# Patternbasierter Ansatz zur Entwicklung von verteilten Krankenakten

Kernbestandteil einer Pattern-Sprache zur Beschreibung und Entwicklung von Softwaresystemen für verteilte Krankenakten


Inaugural-Dissertation zur Erlangung der Doktorwürde der Philosophischen Fakultät III (Sprach-, Literatur- und Kulturwissenschaften) der Universität Regensburg

vorgelegt von

**Wolfgang Wiedermann**

aus Floß

2013

**Erstgutachter:** Prof. Dr. phil. Christian Wolff
**Zweitgutachter:** Prof. Dr. rer. soc. Rainer Hammwöhner

# Inhaltsverzeichnis

# 1. Einleitung

Im Gesundheitswesen setzt sich immer stärker eine auf unterschiedliche Institutionen verteilte Arbeitsweise durch (Rienhoff 2004). Leistungen innerhalb eines Behandlungsprozesses werden von verschiedenen, jeweils spezialisierten Anbietern erbracht. Das lässt sich vor allem durch den gestiegenen Zwang zu kosteneffizienter Arbeitsweise begründen. Besonders Eingriffe und Behandlungsschritte, die extrem hoch qualifiziertes Personal und teure Medizintechnik erfordern, werden so an wenigen, dafür aber gut ausgerüsteten Behandlungszentren durchgeführt. Deren Kapazität kann durch eine Reduktion der durchschnittlichen Verweildauer der Patienten erhöht werden, sodass die Fallzahlen und somit auch die Einnahmen bei gleichbleibenden Ressourcen steigen. Diese Reduktion der Verweildauer ist wiederum nur dann möglich, wenn andere Institutionen die vor und nach der Erbringung der zentralen Leistung nötigen Behandlungsschritte durchführen. Auf diese Weise werden in zunehmendem Maße medizinische Dienstleistungen durch einen Verbund mehrerer Institutionen ausgeführt.

Durch die immer stärker wachsende Zahl institutionsübergreifender Gesundheitsdienstleistungen steigt auch die Notwendigkeit einer gemeinsam verwendbaren Dokumentation des Behandlungsprozesses (Haggerty u. a. 2003; Gulliford u. a. 2006). Folglich wird eine Systemarchitektur oder die Architektur eines Systemverbunds gesucht, die es ermöglichen, dass alle Beteiligten ihren Anteil am Behandlungsprozess gemeinsam dokumentieren und die für sie relevante Dokumentation vorangegangener Schritte einsehen können.

Die bisherigen Krankenhausinformationssysteme (KIS) werden diesen Anforderungen aus verschiedenen architekturbedingten Gründen nicht gerecht (Schwarze u. a. 2005). So übersteigen beispielsweise die datenschutz- und sicherheitsbedingten Anforderungen deutlich jene, die für Systeme gelten, die ausschließlich innerhalb eines Krankenhauses betrieben werden. Außerdem stellen die stark abgeschotteten Netzwerke der einzelnen Institutionen Grenzen dar, die mit den Mitteln aktueller KIS nur schwer überwunden werden können.

Auch die derzeit entstehenden oder existierenden Lösungsansätze zur Bewältigung dieses Problems scheinen aus unterschiedlichen Gründen von deren Zielgruppe nur schlecht angenommen zu werden (Bayer u. a. 2007). Zum einen ist die Bedienung oft wenig intuitiv und wird häufig durch medienbruchbedingte Wechsel zwischen diversen Bedienoberflächen zusätzlich erschwert[1]. Zum anderen decken die Systeme oftmals nur einen Teil der für den Benutzer relevanten Partner-Institutionen ab. Insgesamt ist eine Integration in den bestehenden Arbeitsprozess oft aus technischen, zumeist durch die Systemarchitektur bedingten Gründen nicht möglich.

Diese architekturbedingten Gründe für gute oder schlechte Integrationsfähigkeit der Softwarelösungen gilt es in Zukunft zu identifizieren, um daraus Handlungsempfehlungen für die Entwicklung problemadäquater Systeme abzuleiten. Eine wesentliche Voraussetzung hierfür ist allerdings die Möglichkeit, Mengen von Systemen so zu beschreiben, dass diese bezogen

[1] Beispiele: TempoBy und Soarian Integrated Care

auf die Bestandteile ihrer Software- und Anwendungsarchitektur vergleichbar und gruppierbar werden.

Derzeitige Untersuchungen, die sich mit den Problemen und der Weiterentwicklung verteilter Krankenakten beschäftigen, betrachten die Probleme vorzugsweise aus der Anwendersicht und auf phänomenaler Ebene. Diese Form der Untersuchung liefert durchaus Hinweise auf die Existenz von Problemen; Sie genügt jedoch nicht, um daraus dauerhafte und möglichst allgemeingültige Handlungsanweisungen für die Entwicklung geeigneter Software-Architekturen abzuleiten. Dazu ist eine stärker an den kausalen Zusammenhängen zwischen innerer Struktur und praktischer sowie dauerhafter Verwendbarkeit (im gegebenen Anwendungskontext) der Software interessierte Form der Untersuchung notwendig.

Andererseits sollte nicht vergessen werden, dass die Medizininformatik ein Teilgebiet der Informatik ist und die erwähnten Integrationsprobleme nicht auf verteilte Krankenakten beschränkt sind. So gibt es in anderen Teil-Disziplinen der Informatik bereits erprobte Vorgehensweisen zur Lösung der dort in ähnlicher Form vorliegenden Probleme. Diese können, etwas modifiziert und an die Gegebenheiten der Medizininformatik angepasst, bereits ein Grundlage zukünftiger Handlungsempfehlungen für die Entwicklung verteilter Krankenakten bieten.

Wesentlich ist hierbei das Fachgebiet *Verteilte Systeme*, das sich seit einigen Jahrzehnten mit der Entwicklung großer Systeme auf verteilten Hardwareinfrastrukturen befasst. Ein Teilgebiet davon, die Enterprise-Application-Integration (EAI), beschäftigt sich auf Basis der Methoden zur Entwicklung verteilter Systeme mit der Integration schon bestehender Systeme.

Ein weiteres für die Entwicklung verteilter Krankenakten relevantes Fachgebiet der Informatik ist das Software Engineering. Es setzt sich damit auseinander, die Entwicklung komplexer Systeme durch die Einführung systematischer Vorgehensweisen sowie abstrahierender und darstellender Methoden zu vereinfachen, um diese Systeme weniger fehleranfällig zu gestalten. Die Möglichkeiten des Software Engineerings, das sowohl für die Beherrschbarkeit großer Softwareprojekte als auch komplexer Zusammenhänge essenziell ist, werden in der Medizininformatik zu selten und in zu geringem Umfang ausgeschöpft (Glesner u. a. 2008).

Ziel der hier vorliegenden Arbeit ist es, ein Konzept zu entwickeln, das unter Zuhilfenahme von Erkenntnissen aus der Forschung über Software Engineering und zu verteilten Systemen bereits zum jetzigen Zeitpunkt eine wesentliche Unterstützung bei der Entwicklung verteilter Krankenakten bietet. Es soll auch als verlässliches Hilfsmittel bei Untersuchungen von bereits entwickelten Systemen verteilter Krankenakten verwendet werden können, sodass diese Untersuchungen auch auf die kausalen Zusammenhänge zwischen nutzerrelevanten Eigenschaften und softwaretechnischen Architektur- und Design-Entscheidungen fokussiert sind.

Die hierfür verwendbaren Methoden des Software Engineering reichen vom systematischen Requirements Engineering (siehe z. B. S. Robertson & J. Robertson 2006) über die modellgetriebene Entwicklung und die domänenspezifische Modellierung (siehe z. B. S. Kelly & Pohjonen 2009; Hen-Tov u. a. 2009; Iseger 2010) bis hin zu den Architektur- und Design-

Patterns. Alle diese Ansätze zielen auf die Bildung abstrahierender Modelle zu komplexen konkreten Problemen ab. Im Laufe der Untersuchung haben sich Patterns als geeignetes Mittel erwiesen, die beiden gefragten Aspekte gemeinsam zu erfüllen. Patterns sind als abstrakte Beschreibungen erprobter Teillösungen einerseits geeignet, Entwickler bei der Konzeption und Realisierung verteilter Krankenakten zu unterstützen, und sie formalisieren andererseits die Beschreibung der Architektur und der Bestandteile verteilter Krankenakten-Systeme. Die Verwendung von Patterns zur Beschreibung der Software und ihrer Architektur hat sich in verschiedenen Projekten bewährt (vgl. z. B. R. E. Johnson 1992). Sie verbessert die Vergleichbarkeit und schafft damit die Grundlage für die systematische Untersuchung größerer Mengen unterschiedlicher Systeme aus dem gleichen Anwendungsgebiet.

Inhaltlich ist die vorliegende Arbeit in fünf weitere Kapitel gegliedert, gefolgt von einem Fazit, das die gewonnenen Erkenntnisse und offenen Fragen zusammenfasst. Kapitel 2 beinhaltet eine systematische Analyse des Begriffs *Verteilte Krankenakte,* einschließlich seines Umfelds und der zum Verständnis des Themengebiets notwendigen Begriffe und Standards aus der medizinischen Informatik. Durch die am Ende des Kapitels zu findende Untersuchung einer Menge, als verteilte Krankenakten klassifizierbarer Softwaresysteme ergibt sich zudem einen Überblick über die unterschiedlichen Ausprägungen verteilter Krankenakten. Davon ausgehend beschreibt Kapitel 3 die Auswahl eines geeigneten Grundkonzepts für eine allgemein verwendbare Entwicklungsmethodik für verteilte Krankenakten und identifiziert die Konzepte Pattern und Pattern-Sprache als geeignet. Kapitel 4 erläutert ein Konzept, das basierend auf den in Kapitel 2 identifizierten Anforderungen, die aus der Anwendungsdomäne der verteilten Krankenakte resultieren, eine solche Pattern-Sprache strukturieren kann. Dieses Konzept wird abschließend durch ein softwaretechnisch umsetzbares Metamodell beschrieben. Kapitel 5 befasst sich mit dem aus 220 Patterns bestehenden Kernbestandteil der Pattern-Sprache für verteilte Krankenakten. Dabei handelt es sich um ein Pattern-System, das entsprechend dem Strukturkonzept aus Kapitel 4 gegliedert ist und sich an den Anforderungen aus Kapitel 2 und 3 orientiert. Der zugehörige Patternkatalog ist in Anhang A: Pattern- und Beziehungskatalog abgebildet. Kapitel 6 gibt das zugehörige Konzept zur anforderungsbezogenen Patternauswahl wider und zeigt seine praktische Anwendung anhand des Einweiserportals des Universitätsklinikums Regensburg.

# 2. Verteilte Krankenakten

Der Begriff *Verteilte Krankenakte* wird in dieser Arbeit verwendet, um verschiedene softwaretechnische, den Verteilungsaspekt betreffende Lösungskonzepte sowohl für elektronische Krankenakten als auch für elektronische Patientenakten (Hans-Ulrich Prokosch 2001) unter einem einheitlichen Überbegriff beschreiben und betrachten zu können.

Hierzu ist der Begriff *Verteilte Krankenakte* wie folgt definiert:

Unter einer verteilten Krankenakte versteht man eine elektronische Kranken- oder Patientenakte, deren Inhalt über mehrere physikalisch und logisch getrennte Systeme verteilt gespeichert ist, die aber dennoch als eine zusammengehörige Akte in Erscheinung tritt. Es handelt sich dabei folglich nicht um eine beliebige Menge von Systemen, die Daten über Patienten verwalten, sondern um eine Menge von Systemen, die miteinander verbunden sind und einen gemeinsamen Patientenkontext oder einen Mechanismus zur virtuellen Erreichung eines gemeinsamen Patientenkontexts besitzen. Diese Menge von Systemen kann anders formuliert auch eine als kohärentes verteiltes System realisierte Kranken- oder Patientenakte genannt werden. Um die Eigenschaften einer Akte zu erfüllen, muss eine verteilte Krankenakte verschiedene Dokumente verwalten.

Idealtypisch verwenden mindestens die Benutzer aus einer organisatorischen Einheit das System über eine gemeinsame einheitliche Benutzeroberfläche, die den Zugriff auf eine Vielzahl von Subsystemen, welche auch in anderen Institutionen oder organisatorischen Einheiten untergebracht sein können, transparent gestaltet.

Zur Verdeutlichung der Idee verteilter Krankenakten werden in den folgenden beiden Abbildungen die zwei elementaren Ausprägungsformen dargestellt und erläutert. Die Unterscheidung der Ausprägungsformen erfolgt dabei sowohl auf Basis der grundlegenden Kommunikationsformen in verteilten Systemen als auch auf Basis der vorliegenden Redundanz der in den beteiligten Subsystemen enthaltenen Daten.

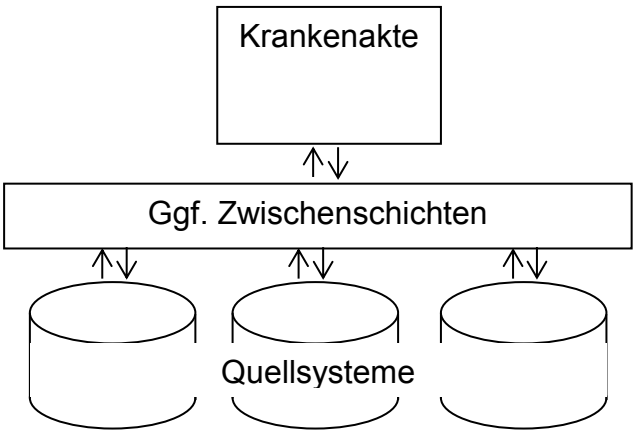



Abbildung 1: Schematische Darstellung einer verteilten Krankenakte mit synchronem Zugriff auf die Quellsysteme

Das Konzept des in der Abbildung 1 dargestellten Systems gründet auf synchronem, anfragebasiertem Zugriff auf die Quellsysteme und einer nichtredundanten Datenhaltung. Im Softwaresystem des Endanwenders tritt die Krankenakte wie eine aus einer einheitlichen und zentral gespeicherten Datenbasis entstandene Akte in Erscheinung. Das bedeutet, die Vielzahl der beteiligten Quellsysteme bleibt für den Endanwender transparent. Technisch realisiert wird das in dieser Ausprägungsform mittels synchroner Remote-Kommunikation, also z. B. mittels Webservice- oder RPC-Aufrufen. Dabei werden die Anfragen durch eine in geeigneter Weise realisierte Zwischenschicht an die jeweils richtigen Quellsysteme weitergeleitet.

Das in Abbildung 2 dargestellte System zeigt eine andere Form, gemäß der eine verteilte Krankenakte gestaltet sein kann. Es verwendet allerdings im Gegensatz zu dem vorher beschriebenen System asynchrone nachrichtenbasierte Kommunikation und setzt auf redundante Datenhaltung.

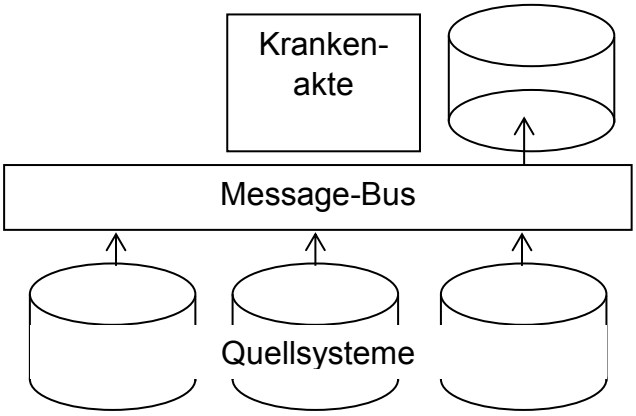



Abbildung 2: Schematische Darstellung einer verteilten Krankenakte, basierend auf nachrichtenorientierter asynchroner Kommunikation

Die Kernkomponente dieses Systemtyps ist ein Message-Server, oft auch Message-Bus genannt, der zwischen den beteiligten Systemen Nachrichten asynchron weiterleitet. Die Quellsysteme versenden bei Veränderungen in ihren Stamm- und Bewegungsdaten gemäß dem Push-Prinzip Nachrichten unter Zuhilfenahme des Message-Servers an das Softwaresystem der Krankenakte, das damit seine eigene Datenbasis zu der des entsprechenden Quellsystems synchron hält.

In beiden dargestellten Fällen ist die Remote-Kommunikation ein wesentlicher Bestandteil des Krankenakten-Systems und begründet so die Zuordnung zur Klasse der verteilten Krankenakten.

## 2.1. Grundlagen und Begriffe

In dieser Arbeit wird eine spezielle Form medizinischer Informationssysteme behandelt. Um die Erläuterungen zu dieser selbst und dem sie umgebenden fachlichen Rahmen in eindeutiger Weise zu verstehen, ist ein gemeinsames Verständnis zentraler Begriffe notwendig. Zur Erreichung dieses Zieles sind in den nachfolgenden Absätzen wichtige Begriffe definiert und falls nötig mit Beispielen beschrieben.

### 2.1.1. Krankenakte

Die Grundlage der medizinischen Dokumentation ist die Krankenakte. Sie ist eine Sammlung medizinischer Dokumente zu einem Patienten und kann entweder als klassisch papiergebundene konventionelle Krankenakte oder als elektronische Krankenakte (siehe 2.1.2) realisiert sein.

Der übergeordnete Begriff zur Krankenakte ist *Akte*. Eine Akte ist, allgemein betrachtet, eine geordnete Sammlung von zu etwas oder zu einer bestimmten Person gehörigen Dokumenten. Dabei ist ein Dokument ein Schriftstück in papiergebundener oder elektronischer Form einschließlich ergänzender Angaben, die für das Verständnis der im Schriftstück enthaltenen Informationen notwendig sind (BMI 2001:§3).

Laut Haas (Haas 2009:S. 117 ff.) gibt eine ordnungsgemäß geführte Krankenakte jederzeit über den Stand der Behandlung Auskunft. Dabei dient die Akte sowohl zur Dokumentation der durchgeführten medizinischen Handlungen, der Durchführungszeitpunkte und der Durchführenden als auch zur Dokumentation der Gründe für die Behandlung und ihrer Ergebnisse. Eine wichtige Eigenschaft aller Dokumente der Akte ist der konsequent dokumentierte Bezug zum behandelten Patienten. Abhängig vom Abdeckungsgrad der gesamten behandlungsrelevanten Daten des zu beschreibenden Patienten spricht man von einer partiellen- oder umfassenden Krankenakte (Leiner u. a. 1999).

### 2.1.2. Elektronische Krankenakte

Eine elektronische Krankenakte ist eine Krankenakte (siehe 2.1.1), die unter Verwendung der elektronischen Datenverarbeitung bzw. der Informationstechnologie (IT) realisiert ist. Sie unterscheidet sich von der konventionellen Krankenakte primär durch die Nutzung digitaler Speichermedien anstelle des Papiers und durch die teilautomatisierte oder automatisierte Verarbeitung der Daten, die an die Stelle der rein manuellen Bearbeitung tritt. Elektronische Krankenakten sind dabei jene IT-technisch realisierten Krankenakten, deren Nutzerkreis die Grenzen eines Krankenhauses oder einer Arztpraxis nicht überschreitet.

Leiner definiert den Begriff *Elektronische Krankenakte* wie folgt:

*„Eine elektronische Krankenakte ist eine umfassende oder partielle Krankenakte, die auf einem elektronischen Datenträger abgelegt ist. In diesem Sinne enthält jedes rechnerbasierte Anwendungssystem zur klinischen Dokumentation zumindest eine partielle elektronische Krankenakte“* (Leiner u. a. 1999:S. 181).

Die am weitesten verbreiteten praktischen Beispiele elektronischer Krankenakten sind Krankenhausinformationssysteme (KIS) und Arztpraxisinformationssysteme (APIS), die üblicherweise als datenbankbasierte Systeme die in Tabellen strukturiert vorliegenden Daten verarbeiten. Aber bereits ein Dokumenten-Management-System, das in Form unstrukturierter textueller Dateien gespeicherte medizinische Dokumente im Patientenkontext verwaltet, ist nach der obigen Definition eine elektronische Krankenakte.

### 2.1.3. Elektronische Patientenakte

Eine elektronische Patientenakte ist die einrichtungsübergreifende Form der elektronischen Krankenakte (siehe 2.1.2) und dient vorrangig der Unterstützung integrierter Versorgungsprozesse. Ein integrierter Versorgungsprozess (Volker Eric Amelung u. a. o. J.) ist ein Prozess der Patientenversorgung, an dem mehrere voneinander unabhängige Einrichtungen des Gesundheitswesens beteiligt sind.

Einrichtungen des Gesundheitswesens sind *„öffentliche und private Einrichtungen, die an der Gesundheitsversorgung der Bevölkerung beteiligt sind“* (Leiner u. a. 1999:S. 181).

Um die Unterscheidung zwischen elektronischer Krankenakte und elektronischer Patientenakte auf Basis ihres auf eine Einrichtung beschränkten oder einrichtungs-übergreifenden Wirkungsbereichs zu begründen, werden anschließend einige verbreitete Definitionen analysiert.

Die folgende aus dem „Gabler Wirtschaftslexikon“ entnommene Definition beschreibt den Unterschied zwischen elektronischer Krankenakte und elektronischer Patientenakte nur unzureichend:

*„Die elektronische Patientenakte ist ein Medium der Informationsspeicherung und Kommunikation.* [Hervorh. durch d. Autor] *Sie erfasst alle Patientendaten (Diagnose, Therapieempfehlungen, unverträgliche Medikamente etc.) in elektronischer Form.“* (Volker Eric Amelung & Axel Mühlbacher o. J.)

Das wesentliche Unterscheidungsmerkmal, nämlich der einrichtungsübergreifende Wirkungsbereich, wird lediglich durch die Erwähnung der Kommunikationsfunktion einer elektronischen Patientenakte angedeutet.

Klarer definiert das Aktionsforum Telematik im Gesundheitswesen den Begriff *Patientenakte*, sodass eine eindeutige Unterscheidung zum Begriff der elektronischen Krankenakte möglich ist:

*„Die elektronische Patientenakte wird hier als eine IT-gestützte, strukturierte Dokumentation verstanden, in der die zeitlich und räumlich verteilt erhobenen Gesundheitsdaten* [Hervorh. durch d. Autor] *eines Menschen zusammengefasst werden. Dies beinhaltet grundsätzlich sämtliche den Patienten wie die Leistungserbringer betreffenden medizinischen und administrativen Behandlungsangaben einschließlich der Prävention. Die Daten werden nach einheitlichen Ordnungskriterien elektronisch erfasst und gespeichert. Diese einrichtungsübergreifende* [Hervorh. durch d. Autor] *elektronische Patientenakte ermöglicht erstmals die problemorientierte Transparenz der Krankengeschichte mit dem Ziel*

*bestmöglicher Versorgung und der Minimierung unerwünschter Belastungen, Verzögerungen und Doppelleistungen"* (Boeske u. a. 2004:S. 9).

Diese Definition stellt unmissverständlich den einrichtungsübergreifenden Kontext, die räumlich und zeitlich verteilte Erhebung und Verwendung der medizinischen Dokumentation, die den Unterschied zwischen Kranken- und Patientenakte bedingen, in den Vordergrund.

Ähnlich wird diese Unterscheidung auch in einem Aufsatz von Prokosch (Hans-Ulrich Prokosch 2001) verwendet, allerdings unter Bezugnahme auf die englischsprachige Definition von Waegemann. Dabei setzt er den deutschen Begriff der elektronischen Krankenakte mit dem des *Electronic Medical Record* (EMR) und den der elektronischen Patientenakte dem Begriff *Electronic Patient Record* (EPR) gleich. Nach der Analyse der Definitionen von Waegemann kommt Prokosch zu dem Schluss, dass nur dann von einer elektronischen Patientenakte gesprochen werden kann, wenn patientenbezogene Daten institutionsübergreifend zusammengeführt werden.

In neueren Publikationen von Waegemann (C. P. Waegemann 2003) wird diese Unterscheidung um weitere untergeordnete Systemtypen wie z. B. den *Computer-based Patient Record* erweitert. Das hier besonders herausgestellte Unterscheidungskriterium zwischen Krankenakte und Patientenakte bleibt aber auch in diesen Systemtypen erhalten.

### 2.1.4. Medizinisches Informationssystem

Dem Begriff der elektronischen Krankenakte und der elektronischen Patientenakte logisch übergeordnet ist der Begriff des medizinischen Informationssystems. Er umfasst die Gesamtmenge aller Daten verarbeitenden Systeme sowie deren sozio-technisches Umfeld, sofern diese eine Beziehung zur medizinischen Tätigkeit aufweisen.

Typische Vertreter medizinischer Informationssysteme sind

- Krankenhausinformationssysteme,
- Praxisinformationssysteme,
- Röntgeninformationssysteme oder
- medizinische Abteilungsinformationssysteme.

Ein medizinisches Informationssystem ist ein Informationssystem, dessen primäre Aufgabe die Unterstützung von Prozessen medizinischer Versorgungseinrichtungen ist.

Leiner definiert hierzu den Begriff *Informationssystem* wie folgt:

Ein Informationssystem ist *„das Teilsystem eines Unternehmens (z. B. einer medizinischen Versorgungseinrichtung), das aus den informationsverarbeitenden Verfahren und den in ihnen beteiligten menschlichen und maschinellen Handlungsträgern in ihrer informationsverarbeitenden Rolle besteht.*

*Die informationsverarbeitenden Verfahren kann man sich als eine Menge formeller und informeller Tätigkeiten und Abläufe vorstellen, nach denen die Informationen verarbeitet und ausgetauscht werden"* (Leiner u. a. 1999:S. 188).

Ein medizinisches Informationssystem ist folglich ein Informationssystem, das sich mit der Erfüllung oder Unterstützung medizinischer Aufgaben befasst. Dieses Verständnis von medizinischen Informationssystemen ist auch mit der folgenden Definition des Begriffs *Informationssystem* vereinbar.

*„Informationssystem: Alle Einrichtungen, Handlungen und Vorschriften der Erfassung, Verarbeitung und Verwertung von Daten/Informationen; Kombination von Rechnern und Anwendungen, die Daten in Informationen umwandeln und Daten, sowie Informationen speichern. I. muss so beschaffen sein, daß*[sic!] *es Daten strukturiert und ordnet, um das Informationsbedürfnis aller Beteiligten am richtigen Ort, zur richtigen Zeit und in der richtigen Verdichtung zu befriedigen. Das Informationssystem dient der Unterstützung administrativer sowie dispositiver Aufgaben und soll strategische Planung und Entscheidungen optimieren helfen"* (Zilahi-Szabó 1995:S. 277).

Ähnlich wie bei den medizinischen Informationssystemen beinhaltet der Begriff des betrieblichen Informationssystems ebenfalls eine Spezialisierung des allgemeinen Begriffs *Informationssystem*. Verschiedene Bestandteile aus der folgenden Definition betrieblicher Informationssysteme kommen auch in medizinischen Einrichtungen vor. Beispiele hierfür sind Buchhaltungssysteme oder Systeme zur Verwaltung von Lagerbeständen. Sie dienen der Erfüllung klassisch betriebswirtschaftlicher Aufgaben in einem Betrieb, dessen Zweck die Erbringung medizinischer Leistungen ist, und sind nicht Bestandteil des medizinischen Informationssystems.

*„Betriebliche Informationssysteme (auch: „Anwendungssysteme") unterstützen Anwender in Wirtschaft und Verwaltung bei der Bewältigung ihrer Aufgaben. Von technischen Systemen (etwa Systemprogramme, Betriebssysteme) unterscheiden sie sich vor allem durch ihre Anwender, die in der Regel nicht Berufe der Informatik oder solche aus deren Nähe ausüben, sondern betriebswirtschaftliche Rollen wie Buchhalter, Disponent, Vertriebsmitarbeiter, Produktionsplaner einnehmen. Zudem dienen betriebliche Informationssysteme der Verwaltung und Steuerung betrieblicher Ressourcen wie etwa Material, Finanzmittel, Fertigfabrikate. Informationssysteme bilden die Leistungs- und Zahlungsströme in einem Unternehmen und zwischen Unternehmen ab und dienen der Steuerung und Überwachung dieser Ströme"* (Uwe Schneider & Dieter Werner 2007:S. 696).

Schnittstellen zwischen dem medizinischen- und betrieblichen Informationssystem gibt es im Bereich der Abrechnung medizinischer Leistungen, der Anforderung von Medikamenten im Rahmen dieser medizinischen Leistungen etc. In diesen Bereichen befinden sich die Schnittstellen zwischen dem betrieblichen und dem medizinischen Bestandteil des gesamten Informationssystems einer medizinischen Einrichtung.

### 2.1.5. Softwaresystem

Ein Softwaresystem ist der Teil eines Informationssystems, der mittels Software realisiert ist. Es handelt sich somit um eine Teilmenge eines Informationssystems. Technisch betrachtet ist ein Softwaresystem Software, die aus einer Menge von programmiertechnischen Bausteinen besteht, wobei das Zusammenwirken dieser Bausteine zur Erledigung von Aufgaben mittels eines oder mehrerer Computer genutzt wird.

Der Begriff *Software* steht als Oberbegriff über der Menge der Softwaresysteme.

Lassmann definiert den Begriff *Software* folgendermaßen:

*„Software ist ein Sammelbegriff für die Gesamtheit der Programme, die zugehörigen Daten und die notwendige Dokumentation, die es erlauben, mit Hilfe des Computers Aufgaben zu erledigen“* (Lassmann 2006:S. 127).

Der Begriff des Softwaresystems basiert auf dem der Software, schränkt dessen Geltungsbereich allerdings auf die Gesamtheit der Programme ein, wie die folgende Definition zeigt.

Ein Softwaresystem ist *„Software, die aus mehreren Bausteinen besteht; die Bausteine bezeichnet man je nach Betrachtungsebene als Programme oder als Module.*

*1. Programmiertechnisch (Programmierung) gesehen besteht ein Softwaresystem aus Programmen (Hauptprogrammen, Unterprogrammen), aus deren Zusammenwirken sich die Lösung eines Problems ergibt.*

*2. Konzeptionell betrachtet besteht ein Softwaresystem aus Modulen, die bei einer Zerlegung nach Abstraktionsprinzipien (Abstraktion) entstehen“*. (Lackes & Siepermann o. J.)

Die Daten und die Dokumentation werden also, für sich betrachtet, zwar als Software, aber nicht als Softwaresystem bezeichnet. Die Anordnung der in den obigen Definitionen genannten Bausteine innerhalb des Softwaresystems wird als Software-Architektur (GI 2010) bezeichnet. Abhängig von der für das Softwaresystem gewählten Architektur und dem gewählten Programmierparadigma variiert auch die Art der für ihren Aufbau verwendeten Bausteine.

Bei der heute nicht mehr üblichen Entwicklung eines ausschließlich auf prozeduralem Code basierenden Softwaresystems sind Funktionen und Prozeduren, wie in der obigen Definition durch die Begriffe *Hauptprogramme* und *Unterprogramme* angedeutet, die Bausteine des Systems. Wird das Softwaresystem rein objektorientiert entwickelt, so sind die Bausteine des Systems Klassen, Objekte und deren Methoden. Eine Form der in dieser Arbeit häufiger referenzierten Bausteine sind Software-Komponenten, die in 2.1.6 näher definiert werden. Sie finden in komponentenbasierten Software-Architekturen (Heineman & Councill 2001) Anwendung.

### 2.1.6. Software-Komponente

Ein weiterer im Rahmen dieser Arbeit verwendeter Begriff ist der der Software-Komponente. Die Verwendung des Begriffs *Komponente* wird allerdings durch seinen vielfältigen, in der Bedeutung unscharfen Gebrauch sowohl in der Alltagssprache als auch im Software-Engineering erschwert (Heineman & Councill 2001:S. 5). Um Mehrdeutigkeiten in den nachfolgenden Ausführungen vorzubeugen, wird ausgehend von frühen Publikationen zu Komponenten und unter Zuhilfenahme von Definitionen aus der aktuellen Fachliteratur eine für diese Arbeit eindeutige Begriffsdefinition für Software-Komponenten erarbeitet.

Die Idee der Bildung von Software-Komponenten wurde auf der NATO-Software-Engineering-Conference im Jahre 1968 erstmals von Douglas McIlroy[2] in dem Beitrag *„Mass Produced Software Components“* (McIlroy 1968) publiziert. Dabei beschreibt er die Idee von Software-Komponenten, stark geprägt vom Denken der damals vorherrschenden strukturierten und prozeduralen Programmierparadigmen, als wiederverwendbare Routinen, die durch verschiedene Maschinen und Benutzer eingesetzt werden können und in Produkt-Familien für beliebige Anwendungsbereiche, qualitativ gruppiert nach Präzision, Robustheit und Performanz, verfügbar sind. Dabei nimmt er die in der Phase der Industrialisierung entstandene Fertigung industrieller Produkte aus Baugruppen, den sogenannten Komponenten, zum Vorbild für eine, aus damaliger Sicht zukünftige Methode der Software-Produktion.

Diese initiale Definition von Douglas McIlroy benennt bereits einige der wesentlichen Eigenschaften, die auch für moderne Software-Komponenten gelten. Dennoch ist für eine exakte Abgrenzung moderner Komponenten von anderen modularisierenden Konstrukten des Software-Engineerings eine weitere Einschränkung der Definition durch zusätzliche notwendige Eigenschaften erforderlich.

Johannes Sametinger liefert in „Software Engineering with Reusable Components“ die folgende Definition:

*„Reusable software components are self-contained, clearly identifiable artifacts that describe and/or perform specific functions and have clear interfaces, appropriate documentation and a defined reuse status“* (Sametinger 1997:S. 65).

Die Analyse von Sametingers Definition aus dem Jahre 1997 erbringt die folgenden Kriterien für Software-Komponenten:

- Wiederverwendbarkeit,
- Abgeschlossenheit,
- eine bestimmte, klar benennbare funktionale Aufgabe,
- eine definierte Schnittstelle und
- geeignete Dokumentation.

[2] Näheres zur Person siehe: http://www.cs.dartmouth.edu/~doug/

Die dabei getroffenen Forderungen nach Wiederverwendbarkeit, Abgeschlossenheit und einer klar benennbaren funktionalen Aufgabe befinden sich bereits in der ursprünglichen Definition von McIlroy aus dem Jahre 1968. Hinzu kommt die Forderung nach einer definierten Komponenten-Schnittstelle, die die Komplexität der Komponente nach außen hin verbirgt und dadurch ihre Benutzung vereinfacht. Außerdem trägt Sametinger in seiner Definition dem Umstand Rechnung, dass sich in der Praxis die fehlende Dokumentation auf die Wiederverwendbarkeit der Komponenten negativ ausgewirkt hat.

Heinemann und Councill definieren im Jahr 2001 Software-Komponenten anhand anderer Kriterien als die beiden oben zitierten Autoren:

*„A software component is a software element that conforms to a component model and can be independently deployed and composed without modification according to a composition standard“* (Heineman & Councill 2001:S. 7).

Zur Interpretation dieser Definition ist die Kenntnis der Definition des darin enthaltenen Begriffs “component model” notwendig.

*„A component model defines specific interaction and composition standards. A component model implementation is the dedicated set of executable software elements required to support the execution of components that conform to the model“* (Heineman & Councill 2001:S. 7).

Diese Definition erweitert die bisher gefundenen Kriterien um die Notwendigkeit eines Komponentenmodells, unabhängiger Veröffentlichbarkeit resp. Verwendbarkeit, Kompositionsfähigkeit ohne Veränderung der Komponente selbst und um die Notwendigkeit eines Kompositionsstandards. Unter solchen Kompositionsstandards sind beispielsweise Methoden zur Abbildung des Komponenten-Lebenszyklus‘ und für die Reaktion auf Veränderungen des Komponentenstatus‘ zu verstehen.

Unter Berücksichtigung aller dieser Eigenschaften ergibt sich innerhalb der vorliegenden Arbeit die folgende Definition für den Begriff *Software-Komponente*:

Software-Komponenten sind in sich geschlossene, funktional zusammengehörige Einheiten von Software, die durch verschiedene Softwaresysteme über eine die innere Komplexität verbergende Schnittstelle genutzt werden können. Sie sind auf beliebige Systeme verteilbar und/oder entfernt von beliebigen Systemen aus nutzbar. Komponenten mit gleicher Schnittstelle und gleichen Standards können gegenseitig ausgetauscht werden, ohne dass eine Änderung am konsumierenden System nötig ist.

### 2.1.7. Verteiltes System

Ein verteiltes System ist aus einer Menge von Komponenten aufgebaut, die sich auf verschiedenen, über existierende Kommunikationswege verbundenen Systemknoten befinden. Zur Erfüllung ihrer Aufgaben müssen diese Komponenten miteinander kommunizieren und sind deshalb auf funktionierende Kommunikationsmechanismen angewiesen. In der folgenden Betrachtung wird ausschließlich auf verteilte Softwaresysteme Bezug genommen. Das be-

deutet, es handelt sich bei den verteilt betriebenen Komponenten um Software-Komponenten, die über vorhandene Computernetze miteinander kommunizieren. Entsprechend dieser Eingrenzung sind auch die anschließenden Definitionen alleinig auf verteilte IT- und Softwaresysteme fokussiert.

Tanenbaum und van Steen definieren den Begriff *Verteiltes System* wie folgt:

*„Ein verteiltes System ist eine Menge voneinander unabhängiger Computer, die dem Benutzer wie ein einzelnes, kohärentes System erscheinen“* (Tanenbaum & Van Steen 2003:S. 18).

Aus einem ebenfalls in der Lehre über verteilte Systeme häufig eingesetzten Standardwerks von Coulouris, Dollimore und Kindberg stammt die folgende Definition:

*„Bei einem verteilten System arbeiten Komponenten zusammen, die sich auf vernetzen Computern befinden und ihre Aktionen durch den Austausch von Nachrichten koordinieren.“* (Coulouris u. a. 2001:S. 17).

Während die Definition von Tanenbaum ihren Fokus auf die in einem Netzwerk verbundenen, aber physikalisch verteilt existierenden Hardware-Komponenten („Computer“) richtet und sehr deutlich aufzeigt, dass der Verteilungsaspekt in einem verteilten System für den Benutzer transparent sein soll, weist die Definition von Coulouris u. a. den ebenfalls für die Betrachtung verteilter Krankenakten wichtigen Aspekt der auf der vernetzten Hardware befindlichen kommunizierenden Software-Komponenten hin. Basierend auf der Verbindung beider Definitionen ist ein verteiltes System in dieser Arbeit folgendermaßen definiert:

Ein verteiltes System besteht aus einer Menge von vernetzten Computern und den darauf befindlichen Komponenten, die ihre Zusammenarbeit durch entfernte Kommunikation koordinieren und nach außen hin als einzelnes kohärentes System in Erscheinung treten.

Da heute die Qualität und Verfügbarkeit von Computernetzen, wie das Internet und die Intensität seiner Nutzung beweisen, ein sehr hohes Niveau erreicht hat, hat auch der Bedarf, IT-Anwendungen als verteilte Softwaresysteme umzusetzen, deutlich zugenommen. Deshalb kann die Umsetzung elektronischer Krankenakten als verteiltes System als technisch realistisches Ziel betrachtet werden, das durch die fachliche Entwicklung zu immer stärker integrierten Versorgungsprozessen auch von praktischer Notwendigkeit gestützt wird.

### 2.1.8. Kommunikation in verteilten Systemen

Wie bereits in der zugehörigen Definition erwähnt, ist Kommunikation zwischen vernetzten Komponenten ein wesentlicher Aspekt verteilter Systeme. Der Begriff Kommunikation beschreibt allgemein die Übertragung oder den Austausch von Daten oder Information zwischen zwei oder mehreren Entitäten. In der Kommunikationswissenschaft werden diese Entitäten Kommunikanten genannt (Klaus Beck 2007:S. 27).

Kommunikation ist grundsätzlich als Prozess definiert, an dem mindestens zwei Seiten beteiligt sind, die etwas (meist Information) austauschen oder miteinander teilen (Klaus Beck 2007:S. 15). Der Gegenstandsbereich der Kommunikation wird in der Kommunikations-

wissenschaft allerdings explizit auf die Kommunikation zwischen menschlichen Individuen eingeschränkt, während diese Arbeit den Fokus auf Kommunikation zwischen Software-Komponenten richtet. Aus diesem Grund werden alle über die beiden genannten Zitate hinausreichenden, spezifisch für die Kommunikationswissenschaft getroffenen Einschränkungen des Begriffs *Kommunikation* aus der Definition von Kommunikation im Kontext verteilter Krankenakten ausgeschlossen.

Der Kommunikation zwischen technischen Systemen wie Software-Komponenten oder unabhängigen Systemprozessen liegt der Austausch von Nachrichten zugrunde. Dabei wird in dieser Arbeit von Kommunikation gesprochen, wenn mindestens zwei Entitäten Nachrichten austauschen. In einem Softwaresystem liegt immer dann ein technisch bedingter Zwang zur Kommunikation vor, wenn mehr als ein Prozess an der Erfüllung einer Aufgabe beteiligt ist, also zwischen diesen unabhängigen Prozessen (z. B. zur Koordination) ein Austausch von Nachrichten nötig ist.

Es gibt zwei grundlegende Formen der Kommunikation, nämlich die asynchrone Kommunikation, bei der zwischen dem Senden der Nachricht und deren Beantwortung eine nicht festgelegte Zeitspanne (die größer als null Zeiteinheiten ist) vergehen kann und die synchrone Kommunikation, bei der der Empfänger die vom Sender kommende Nachricht sofort erhält, verarbeitet und beantwortet. Die Unterschiede zwischen den beiden genannten Kommunikationsformen lassen sich mittels zweier Beispiele für Kommunikation aus dem alltäglichen Leben verdeutlichen.

**Beispiel: asynchrone Kommunikation**

Ein klassisches Beispiel für asynchrone Kommunikation ist der Briefverkehr. Dabei schickt der Sender seine Nachricht unabhängig von der Verfügbarkeit bzw. Empfangsbereitschaft unter Zuhilfenahme der Post als Übertragungsdienst ab und hofft, dass diese den Empfänger erreicht. Ab dem Zeitpunkt des Absendens der Nachricht kann der Sender anderen Tätigkeiten nachkommen und nach einiger Zeit möglicherweise eine ebenfalls postalische Antwort auf seine Nachricht erhalten.

Der Empfänger prüft in diesem Kommunikations-Szenario regelmäßig die Verfügbarkeit neuer Nachrichten, da diese eintreffen, ohne bei ihm ein explizites Benachrichtigungs-Ereignis auszulösen. Sollte eine neue Nachricht vorhanden sein, so kann er sie zu dem Zeitpunkt bearbeiten, zu dem ihm die dazu benötigte Zeit zur Verfügung steht. Entsteht im Rahmen der Verarbeitung die Notwendigkeit, den Absender der Nachricht über einen geänderten Sachverhalt zu informieren, so kann der Empfänger der ursprünglichen Nachricht auch zum Sender einer Antwort-Nachricht werden. Der Versand und die Verarbeitung der Antwort-Nachricht erfolgen nach dem gleichen Prinzip wie bei der hier bereits beschriebenen Nachricht.

Die im Rahmen dieser Arbeit betrachtete Kommunikation zwischen Software-Komponenten unterscheidet sich lediglich dadurch, dass die kommunizierenden Entitäten nicht Personen, sondern Software-Komponenten und der zustellende Dienst nicht die Post, sondern beispiels-

weise eine Message Queue, also eine Warteschlange für elektronisch zuzustellende Nachrichten sind.

Coulouris, Dollimore und Kindberg beschreiben asynchrone Kommunikation zwischen verteilten Software-Komponenten wie folgt:

*„Bei der asynchronen Form der Kommunikation ist die Verwendung der send-Operation nicht blockierend, sodass der sendende Prozess weiterarbeiten kann, sobald eine Nachricht in einen lokalen Puffer kopiert wurde, und die Übertragung der Nachricht erfolgt parallel zur Arbeit des sendenden Prozesses. Die receive-Operation kann blockierende und nicht blockierende Varianten haben“* (Coulouris u. a. 2001:S. 160).

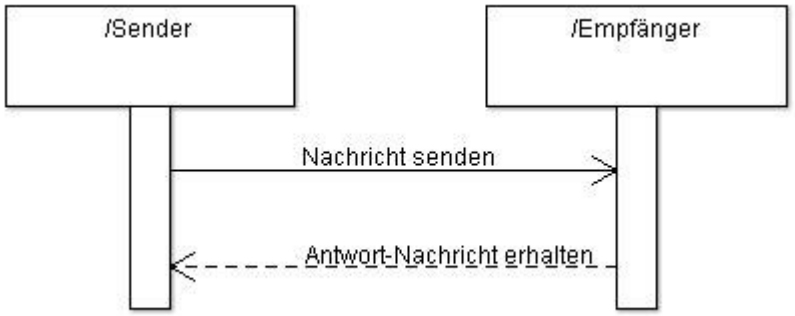



Abbildung 3: asynchroner Nachrichtenaustausch zwischen Sender und Empfänger

In Abbildung 3 ist die asynchrone Kommunikation zwischen den Komponenten „Sender“ und „Empfänger“ in Form eines Sequenzdiagramms der Unified Modelling Language (UML) (OMG 2009:S. 506–510) dargestellt. Es zeigt, dass die Prozesse beider Komponenten durchgehend aktiv sind und nicht durch explizites Warten auf den anderen Kommunikanten unterbrochen werden.

Betrachtet man die Erläuterung von Tanenbaum und van Steen zu verbindungslosen Protokollen (Tanenbaum & Van Steen 2003:S. 78), so lassen sich eindeutige Parallelen zur Definition asynchroner Kommunikation zwischen Software-Komponenten oder -Prozessen feststellen. Folglich sind verbindungslose Protokolle ein mögliches Mittel zur Erreichung asynchroner Kommunikation.

Weitere verbreitete Mittel, die in der Praxis zur Realisierung asynchroner Kommunikation zwischen Software-Komponenten verwendet werden, sind spezielle Messaging-Protokolle wie Java Messaging Service (JMS)(Hapner u. a. 2003) oder Microsoft Message Queuing (MSMQ)(Microsoft o. J.). Diese beiden Beispiele können gemeinsam mit vielen weiteren unter dem Begriff der Message orientierten Middleware zusammengefasst werden.

**Beispiel: synchrone Kommunikation**

Ein alltägliches Beispiel für die synchrone Kommunikation ist das Telefongespräch. Es erfordert einerseits, dass beide Kommunikanten zum Zeitpunkt des Gesprächs an ihrem jeweiligen Telefon verfügbar sind. Außerdem erfordert die erfolgreiche Verständigung, dass

nacheinander gesprochen wird. Auf eine Frage folgt eine Antwort oder eine Rückfrage (vgl. auch Misoch 2006:S. 54).

Zwischen Software-Komponenten wird synchrone Kommunikation über blockierende Request-Reply-Protokolle abgewickelt, bei denen der anfragende Kommunikant auf die Antwort seines Kommunikationspartners wartet und erst danach mit seiner Arbeit fortfährt (vgl. auch Pink u. a. 2002:S. 117). Das bedeutet, dass die Operationen in beiden beteiligten Prozessen synchronisiert, d. h. in einer auch zwischen beiden Kommunikanten definierten Reihenfolge ausgeführt werden. So ist z. B. auch die Anfrage eines Browsers bei einem Webserver eine synchrone Anfrage, da der Browser zur Erledigung seiner Aufgabe, dem Visualisieren einer Webseite, die Antwort des Webservers, der ihren Inhalt liefert, abwarten muss.

Auf höherer Ebene sind entfernte Prozedur-Aufrufe (RPC, siehe Srinivasan 1995; Sun Microsystems 1988; White 1976) oder Methoden-Aufrufe (RMI, siehe Oracle o. J.) die häufigsten Formen synchroner Kommunikation zwischen Software-Komponenten (Tanenbaum & Van Steen 2003:S. 77). Da die synchrone Kommunikation wegen der Wartezeit im Client-Prozess bei Prozeduren oder Methoden ohne Rückgabewert wenig sinnvoll ist, gibt es laut Tanenbaum und Van Steen (Tanenbaum & Van Steen 2003:S. 100 ff.) auch RPC-APIs, die für Prozeduren ohne Rückgabewert asynchrone Aufrufe erlauben.

Coulouris, Dollimore und Kindberg beschreiben die synchrone Kommunikation zwischen verteilten Software-Komponeten wie folgt:

*„Bei der synchronen Form der Kommunikation werden der sendende und der empfangende Prozess für jede Nachricht synchronisiert. In diesem Fall sind sowohl send als auch receive blockierende Operationen. Immer wenn ein send abgesetzt wird, wird der sendende Prozess blockiert, bis das entsprechende receive abgesetzt wurde. Immer wenn ein receive abgesetzt wird, blockiert der Prozess bis eine Nachricht ankommt“* (Coulouris u. a. 2001:S. 160).

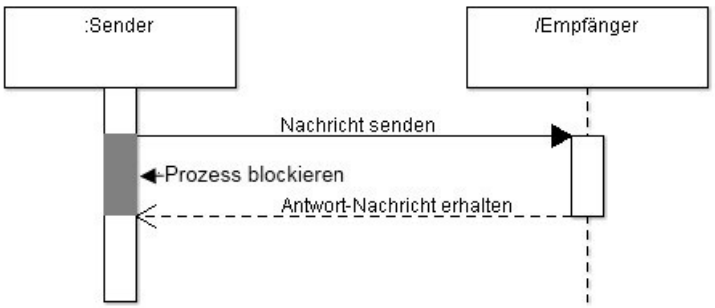



Abbildung 4: synchroner Nachrichtenaustausch zwischen Sender und Empfänger mit wartendem Senderprozess

Abbildung 4 zeigt den Ablauf eines Request/Response-Zyklus synchroner Kommunikation in Form eines Sequenzdiagramms. Bei der Kommunikation zwischen menschlichen Individuen spricht man von synchroner Kommunikation, wenn die Zeit zwischen Senden und Empfang der Nachricht ungefähr gleich null ist. Zwischen Software-Komponenten oder -Prozessen ist analog dazu dann von synchroner Kommunikation zu sprechen, wenn der Senderprozess nach dem Senden der Nachricht bis zum Erhalt der Antwort-Nachricht blockiert ist, sodass für ihn die Antwort die direkte Folge seiner Anfrage ist. Der Sender – in dieser Konstellation

gemeinhin Client genannt – führt also bei einem synchronen Aufruf zwischen Anfrage und Antwort keinen weiteren Code aus.

Abschließend zur Betrachtung asynchroner und synchroner entfernter Kommunikation bleibt noch festzuhalten, dass sowohl mit den erwähnten Werkzeugen zur Realisierung asynchroner Kommunikation auch synchrone Szenarien und mit den Werkzeugen zur Realisierung synchroner Kommunikation auch asynchrone Szenarien realisierbar sind. Das ist allerdings mit einem erheblichen technischen Mehraufwand verbunden, sodass die Eignung der Werkzeuge für den beschriebenen Einsatz eindeutig erkennbar ist.

**Auslösen der Datenübermittlung per Push oder Pull**

Eine weitere wichtige Betrachtungsform von Kommunikation basiert darauf, welche der kommunizierenden Entitäten der Verbraucher der übertragenen Daten ist. Es wird unterschieden, ob der Verbraucher als aktive Entität durch eine Anfrage bei einem Server-Prozess gezielt die Übermittlung von Daten in Auftrag gibt (häufig synchron) oder ob der Verbraucher als passive Entität von einem aktiven Serverprozess Datenpakete erhält (häufig asynchron) (Bodendorf 2008:S. 5 ff.).

Vom Push-Prinzip (vgl. z. B. Haas 2006:S. 45) spricht man, wenn die aktive Entität autonom Daten, die dort möglicherweise gebraucht werden an empfangende passive Entitäten verteilt (siehe Abbildung 5). Der Name des Prinzips ist vom englischen Verb „push“ abgeleitet, das „schieben“ bedeutet, hier aber in der Bedeutung „bringen“ verwendet wird.

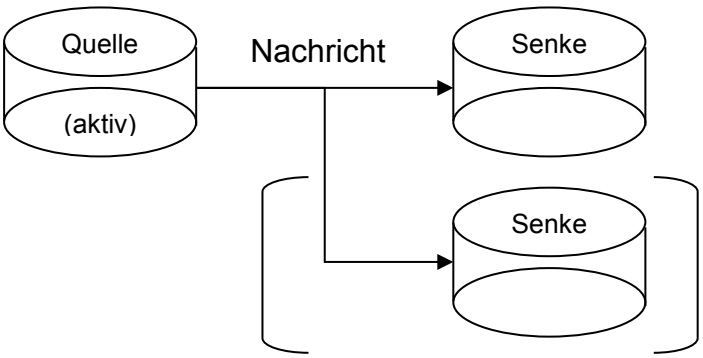



Abbildung 5: Datenfluss beim Push-Prinzip

Beim Pull-Prinzip fragt der Verbraucher aktiv bei einer Server-Entität an und holt somit die benötigten Daten aktiv ab. Sein Name wird vom englischen Verb „pull“ abgeleitet, das „ziehen“ bedeutet, hier aber in der Bedeutung „holen“ verwendet wird.

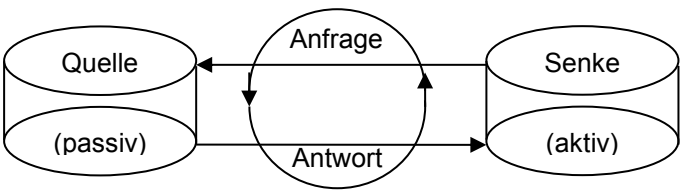



Abbildung 6: Datenfluss beim Pull-Prinzip

## 2.2. Einordnung in den Kontext medizinischer Informationssysteme

Der Begriff des medizinischen Informationssystems stellt einen Sammelbegriff für sämtliche in medizinischen Versorgungseinrichtungen verwendeten IT-Systeme einschließlich ihres soziotechnischen Umfelds dar. Diese Systeme werden genutzt um Patienten-Stammdaten, Falldaten, Verlaufsdaten, Symptome, Diagnosen und Notizen zu verwalten (Haas 2009:S. 276 ff.). Die zentrale Entität, um die alle weiteren dokumentierenden Objekte in medizinischen Informationssystemen angeordnet sind, ist der Patient.

Systeme zur Realisierung verteilter Krankenakten sind eine spezielle Form medizinischer Informationssysteme, deren Abgrenzung von der Restmenge medizinischer Informationssysteme die Fragestellung dieses Kapitels ist. Dazu wird in Abschnitt 2.2.1 auf verschiedene Ansätze zur Klassifikation eingegangen, während deren Verwendbarkeit bzw. Verwendung zur Einordnung verteilter Krankenakten sowie die Definition abgrenzender Kriterien innerhalb dieser Klassifikationsschemata anschließend in Kapitel 2.2.2 behandelt werden.

### 2.2.1. Klassifikation medizinischer Informationssysteme

Medizinische Informationssysteme sind auf der Basis verschiedener Kriterien klassifizierbar. Jedes der hier angeführten Klassifikationskonzepte stellt eine logische Blickrichtung auf die Menge der medizinischen Informationssysteme dar.

**Klassifikation anhand des Anwendungsgebiets**

Die Klassifizierung medizinischer Informationssysteme anhand ihres Einsatz- bzw. Anwendungsgebietes ist einer der möglichen Ansätze. Er trägt der Eigenschaft der Medizininformatik als Disziplin der angewandten Informatik Rechnung. Dazu bietet Haas (Haas 2009:S. 33–34), ausgehend von den derzeit im medizinischen Alltag eingesetzten Systemtypen, folgende Auflistung:

- Krankenausinformationssystem
- Arztpraxisinformationssystem
- Laborinformationssystem
- Radiologisches Informationssystem
- Histologie-Informationssystem
- Picture Archiving and Communication System (PACS)
- Pflegeinformationssystem
- Tumordokumentationssystem
- Operations-Dokumentationssystem
- Betriebsärztliches Dokumentationssystem

Die nähere Betrachtung dieser Auflistung lässt erkennen, dass die hier aufgeführten Gruppen medizinischer Informationssysteme weder abschließend vollständig noch inhaltlich überschneidungsfrei sind. So kann beispielsweise ein Arztpraxisinformationssystem in einer chirurgischen Praxis bereits ein vollständiges Operations-Dokumentationssystem beinhalten

oder eine radiologische Praxis auch ein umfangreich ausgestattetes radiologisches Informationssystem anstelle eines Praxisinformationssystems verwenden. Diese beiden Beispiele zeigen, dass die Übergänge zwischen den oben genannten Informationssystemtypen fließend sind und die Zuordnung eines konkreten Systems nicht immer eindeutig ist. Aus diesem Grund wird die Betrachtung des Anwendungsgebietes ausschließlich zur Klassifizierung medizinischer Informationssysteme auf der Basis ihres fachlichen Schwerpunkts oder ihrer fachlichen Schwerpunkte eingesetzt.

**Klassifikation anhand der Architektur**

Die Klassifikation medizinischer Informationssysteme anhand der ihnen zugrunde liegenden Architektur ist ein weiterer Ansatz. Bei der Betrachtung der Architekturen von IT-Systemen und ihrer organisatorischen Umgebung wird, basierend auf dem Detaillierungsgrad der Betrachtungstiefe, zwischen System-, Informationssystem-, Unternehmens-, IT-System- und Softwarearchitektur unterschieden. Zur Klassifikation medizinischer Informationssysteme werden in dieser Arbeit ausschließlich die Informationssystemarchitektur und die Softwarearchitektur verwendet, da diese für die Einordnung von verteilten Krankenakten in Abschnitt 2.2.2 aufgrund der technischen Orientierung des Untersuchungsgegenstands gut geeignet sind.

Die Betrachtungsweise der gröbsten Granularitätsstufe ist die der Systemarchitektur. Die Systemarchitektur umfasst die Architekturen des System-Designs sowie die Prozesse des zugehörigen Produkt-Lebenszyklus‘ (IEEE 2005:S. 9). Unter Architekturen des Designs werden hier die Funktionalität und den Aufbau des Systems betrachtende Architekturen mit speziellem Fokus auf die Unterstützung eines festgelegten Ziels verstanden. Das Ziel der Software-Architektur hingegen ist die Unterstützung bzw. Strukturierung der Entwicklung eines Softwaresystems innerhalb des durch die Systemarchitektur beschriebenen Gesamtsystems. Sie beschreibt somit die Struktur eines Softwaresystems, also die enthaltenen Software-Elemente einschließlich ihrer nach außen sichtbaren Eigenschaften und der zwischen ihnen existierenden Beziehungen (Bass u. a. 2003:S. 21).

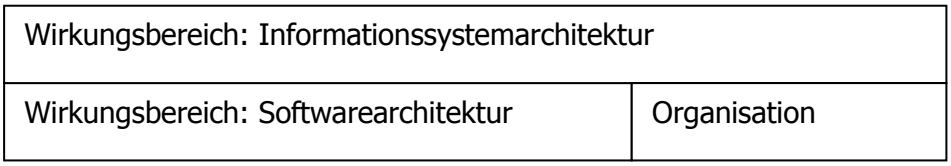



Abbildung 7: Wirkungsbereiche Informationssystem- und Softwarearchitektur

Die Informationssystemarchitektur ist die Systemarchitektur eines Informationssystems, wobei ein Informationssystem, wie in Abschnitt 2.1.4 definiert, ein Softwaresystem einschließlich des zugehörigen organisatorischen und fachlichen Kontextes ist. Folglich sind die Softwarearchitektur und die Aufbauorganisation, wie in Abbildung 7 gezeigt, die in der nächst detaillierteren Betrachtungsebene direkt der Informationssystemarchitektur untergeordneten Architekturtypen.

***Betrachtung nach Gleichartigkeit und Anzahl der Systembestandteile***

Informationssystem- und Softwarearchitekturen können nach der Anzahl und Menge ihrer logischen Bestandteile sowie deren Zusammengehörigkeits- und Gleichartigkeits-eigenschaften unterschieden werden.

Haas (Haas 2009:S. 62–66) unterteilt hierzu die folgenden grundlegenden Systemarchitektur-Typen:

- Monolithisches System;
- Heterogenes System;
- Komponentenbasiertes System.

Er nennt als wesentliche Unterscheidungsmerkmale außerdem die Eigenschaften logische Gliederung der Elemente und Flexibilität des Systems.

Monolithisch oder zentral

Eine monolithische Systemarchitektur ist eine Architektur ohne nach außen hin sichtbare logische Untergliederung, die gleichzeitig eine intensive Kopplung der in ihrem inneren enthaltenen Bestandteile aufweist. Alle wesentlichen Bestandteile des Systems sind zu einem homogenen Gebilde zusammengefügt (Uwe Schneider & Dieter Werner 2007:S. 263).

Fink definiert ein monolithisches IT-System wie folgt:

*„Ein monolithisches IT-System ist als untrennbare Einheit gestaltet („großes Ganzes“). Eine monolithische Architektur folgt keiner expliziten Gliederung in Teilsysteme (Komponenten im weitesten Sinne) und steht damit im Gegensatz zu einem verteilten IT-System.“* (Fink 2009)

Das bedeutet – bezogen auf das gesamte Informationssystem – dass es aus der Sicht seines Betrachters als eine ganze, logisch nicht trennbare Einheit in Erscheinung tritt. Vor allem die Benutzer des Informationssystems nehmen folglich das enthaltene Softwaresystem als ein nicht in logische Subsysteme gegliedertes System wahr. Die Menge der Prozesse der zum Informationssystem gehörigen Organisation muss deshalb in diesem einen zentralen System abgebildet sein, und deren Bedienung muss über eine einheitliche Benutzeroberfläche erfolgen.

Ein Vorteil monolithischer Systeme ist beispielsweise deren hohe Zuverlässigkeit, die aus der Tatsache resultiert, dass durch die Realisierung als zentrales System keine Schnittstellen-Probleme und partiellen Ausfälle möglich sind. Außerdem ist auch kein zusätzlicher Aufwand für Schnittstellenpflege erforderlich. Ein wesentlicher Nachteil ist die schlechte Erweiterbarkeit, die beispielsweise aus den häufig fehlenden offen zugängigen Schnittstellen und der hohen Kohäsion zwischen den Code-Bestandteilen resultiert.

Heterogen

Ein heterogenes System ist ein System, das sich aus uneinheitlichen Elementen zusammensetzt. Es ist das logische Gegenteil eines homogenen Systems. Bezogen auf die Architektur eines Informationssystems bedeutet das, dass die Elemente des Gesamtsystems auf unterschiedlichen Hardwareplattformen untergebracht sind, eine Vielzahl von untereinander nicht kompatiblen Betriebssystemen oder eine große Zahl unterschiedlicher Kommunikationsprotokolle verwendet werden.

Haas schreibt hierzu:

„*Für heterogene Informationssysteme gilt: Die Software für die im logischen Architekturmodell aufgeführten Komponenten stammen von verschiedenen Herstellern, die alle mit eigenen Datenmodellen und Datenhaltungen, Software-Architekturen sowie Funktionen und Benutzeroberflächen arbeiten*“ (Haas 2009:S. 64).

Die Kommunikation in heterogenen Informationssystemen wird häufig über Kommunikationsserver abgewickelt. Sie vermitteln zumeist auf Basis asynchroner nachrichtenorientierter Kommunikation (siehe 2.1.8) zwischen den verschiedenen Elementen des Systems und den unterschiedlichen verwendeten Kommunikationsstandards. Das ist unter anderem deshalb notwendig, um deren redundante Datenbasen aktuell und inhaltlich konsistent zu halten. Die Elemente des Systems werden in der obigen Definition von Haas *Komponenten* genannt.

Die Kopplung zwischen den Bestandteilen eines heterogenen Systems ist lose und erfolgt oft unter Zuhilfenahme von Kommunikationssystemen, die eigens dafür aufgebaut sind, um die Kommunikation zwischen mehreren, nicht aufeinander abgestimmten Subsystemen zu ermöglichen.

Ein wesentlicher Nachteil heterogener Informations- und Softwaresysteme liegt in dem erheblichen Aufwand, der für die Entwicklung von Schnittstellen zur Integration der heterogenen Bestandteile des Gesamtsystems erforderlich ist (Hasselbring 2000). Die Vorteile heterogener Systeme sind beispielsweise in der guten Angepasstheit der Einzelsysteme an die Anforderungen ihres Einsatzbereichs zu finden (Haas 2009:S. 66).

Komponentenbasiert

Ein komponentenbasiertes Informationssystem besteht aus einer Menge von miteinander zu einem kohärenten Informationssystem kombinierbaren Elementen, den Komponenten. Zum Zeitpunkt der Entwicklung einer Komponente ist noch offen, in welchen Kombinationen sie mit anderen Komponenten zu einem System zusammengefügt wird (Uwe Schneider & Dieter Werner 2007:S. 702). Für die softwaretechnischen Bestandteile des Informationssystems gelten die gleichen Annahmen wie für die in Abschnitt 2.1.6 definierten Software-Komponenten.

Komponentenbasierte Systeme kombinieren die Vorteile von monolithischen und heterogenen Systemen (Haas 2009:S. 66). Im Gegensatz zu den Bestandteilen heterogener Systeme besitzen Komponenten je eine eindeutig definierte Schnittstelle und verwenden einen gemeinsamen Kommunikationsstandard. Der Komponenten-Standard stellt Mittel und Methoden zur Verfügung, die eine Kombination von Komponenten zu Systemen ermöglichen. Die Kommunikation zwischen den Komponenten kann entweder synchron oder asynchron realisiert sein. Eine zusätzliche Komponente, die zwischen unterschiedlichen Standards zweier Kommunikationspartner übersetzt, ist aufgrund des gemeinsamen Standards nicht nötig.

Grenzfall serviceorientierte Architektur

Eine weitere häufig genannte Architekturform ist die serviceorientierte Architektur (SOA). Sie lässt sich aufgrund großer Gestaltungsfreiheit nicht eindeutig von den drei bisher genannten Formen abgrenzen und wird deshalb nicht als zusätzliche grundlegende Architekturform verwendet. Abhängig von ihrer konkreten Umsetzung kann eine SOA sowohl als heterogen als auch als komponentenbasiert klassifiziert werden. Die Einordnung hängt primär davon ab, ob innerhalb der konkreten Umsetzung Kommunikation in einer Vielfalt von Standards über einen Enterprise Service Bus (ESB) vermittelt wird, oder die gesamte Kommunikation auf einem einheitlichen Standard, wie z. B. Webservices mit SOAP, basiert.

***Betrachtung anhand der Verteilung der Systembestandteile***

Eine weitere wesentliche Eigenschaft eines mehrbenutzerfähigen Softwaresystems ist die Frage der logischen Verteilung seiner Bestandteile auf einzelne Rechnerkomponenten. Diese Betrachtungsweise wird (in Uwe Schneider & Dieter Werner 2007:S. 226) „horizontale Architektur“ genannt. Da medizinische Informationssysteme in üblichen Versorgungseinrichtungen heute grundsätzlich von mehreren Benutzern parallel genutzt werden, sind diese auch zwangsläufig verteilt realisiert. Art und Umfang dieser Verteilung können allerdings variieren.

Verteilungsbezogene Gruppierung von Softwarearchitekturen:

- Monolith mit hardwaretechnisch verteilter Nutzung (Host mit Terminals);
- Zentrales System mit softwaretechnisch verteilter Nutzung (Client – Server);
- Verteiltes System in Schichtenarchitektur (Drei- oder Mehr-Schichten-Architektur);
- Verteiltes System mit gleichberechtigten Knoten (Peer to Peer).

Host mit Terminals

Das organisatorisch bedingte Minimum der Verteilung ist, dass ein eigentlich vollständig zentrales, auf nur einem Rechner betriebenes System zumindest über seine, auf verteilten Eingabegeräten visualisierten Masken verwendet werden kann.

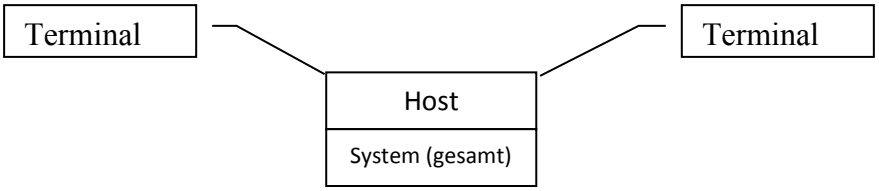



Abbildung 8: Terminals und Großrechner

Bezogen auf die Softwarearchitektur liegt hierbei keinerlei Verteilung der existierenden Softwarekomponenten vor, da alle auf dem gleichen Host ausgeführt werden. Deshalb handelt es sich dabei um die einzige „Verteilungsform“, die mit einer echten monolithischen Softwarearchitektur realisiert werden kann. Abbildung 8 zeigt schematisch zwei Terminals, die es ihren Benutzern ermöglichen, Aufgaben auf einem zentralen Rechner (Host) durchzuführen. Die Terminals werden im Rahmen dieser Aufgabe nicht als Rechner aktiv, sondern dienen lediglich als Ein- und Ausgabegeräte.

Beispiele für diese Form der verteilten Nutzung zentraler Ressourcen sind neben den früheren Kombinationen aus Großrechnern und RS-232-Terminals auch modernere Lösungen, die aus sogenannten Terminalservern und Thin-Clients bestehen (Wendroth 2001).

Client – Server

Die Client-Server-Architektur (Uwe Schneider & Dieter Werner 2007:S. 421) ist die Minimalform eines verteilten Systems. Sie stellt eine Softwarearchitektur mit zwei logischen Schichten, dem Client und dem Server dar. Die Grundlagen zur Kommunikation in verteilten Systemen sind in Abschnitt 2.1.8 hinreichend erläutert. Client-Server-Architekturen können sowohl mit synchronen als auch asynchronen Mitteln der entfernten Kommunikation realisiert werden. Häufig kommunizieren diese Systeme synchron mit einem Datenbank-Management-System.

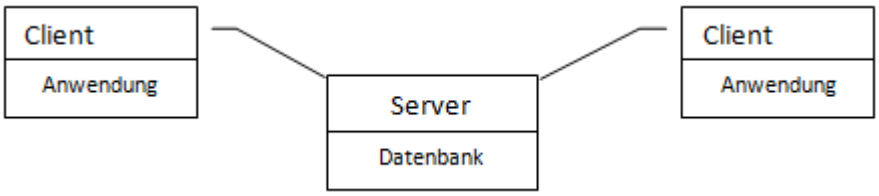



Abbildung 9: Beispiel einer Client-Server-Architektur

Das Architekturschaubild (Abbildung 9) weist eine gewisse Ähnlichkeit zu dem in Abbildung 8 wiedergegebenen auf, allerdings mit dem Unterschied, dass bei einer Client-Server-Architektur mindestens ein gewisser Teil der Anwendungslogik im Client ausgeführt wird, während beim Betrieb eines Systems mit Terminals die Anwendungs- und Darstellungslogik ausschließlich im Server oder Host ausgeführt wird.

Verteiltes System in Schichtenarchitektur

Die logische Weiterführung des Client-Server-Konzeptes, das eine Architektur mit zwei logischen Schichten beinhaltet, ist die Entwicklung von verteilten Systemen in Form von drei-

oder mehrschichtigen Architekturen (Uwe Schneider & Dieter Werner 2007:S. 421). Bei einer Drei-Schichten-Architektur, wie sie in Abbildung 10 dargestellt ist, wird die Anwendungslogik, die bei Client-Server-Systemen üblicherweise im Client untergebracht ist, in einer eigenen logischen Schicht zwischen dem Benutzer-Frontend und der Datenhaltung untergebracht.

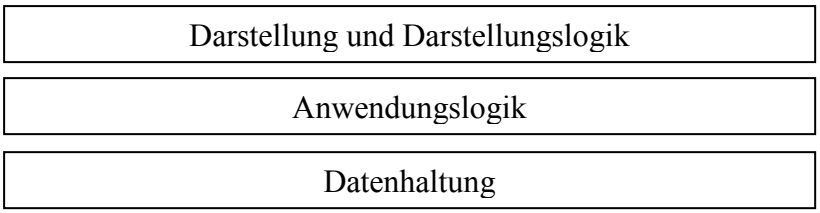



Abbildung 10: Drei-Schichten-Architektur

Bei komplexen Systemen, besonders wenn diese in Form von serviceorientierten Architekturen aufgebaut sind (vgl. z. B. Josuttis 2007:S. 72), hat sich in der Praxis eine Aufteilung auf eine noch größere Zahl von Schichten durchgesetzt. Die zugehörigen Software-Architekturen werden als Multi-Tier-Architekturen, also als mehrschichtige Architekturen bezeichnet.

Abbildung 11 zeigt beispielhaft den Aufbau einer Schichtenarchitektur mit fünf logischen Schichten.


Abbildung 11: Beispiel einer Architektur mit fünf Schichten

Das Prinzip sowie die Vor- und Nachteile der Schichtenarchitektur sind im Anhang auf S. 543 in den Ausführungen zum Schichtenarchitektur-Muster sowie in den dort angegebenen Quellen detailliert beschrieben.

Verteiltes System mit gleichberechtigten Knoten

Im Gegensatz zu den bisher erläuterten Verteilungsmodellen wird bei Peer-to-Peer-Systemen nicht zwischen der Anbieter- und Konsumentenfunktion der Systembestandteile unterschieden. Die Bestandteile des Systems arbeiten folglich als gleichberechtigte Knoten (Peers) sowohl als Konsument als auch als Anbieter von Diensten. (Steinmetz & Wehrle 2004)

Während die Peers zur eigentlichen Nutzung der Dienste direkt miteinander kommunizieren, kann die Suche nach anderen Peers durchaus unterschiedlich realisiert sein. So ist die

dezentrale Ressourcennutzung eine Eigenschaft, die sowohl hybriden als auch reinen Peer-to-Peer-Systemen gemeinsam ist. Bei der Organisation des Peer-to-Peer-Netzes unterscheiden sich reine Peer-to-Peer-Systeme jedoch dadurch von den hybriden Modellen, dass sie auch bei der Selbstorganisation dezentrale Mechanismen verwenden. Solch ein Mechanismus kann beispielsweise das auf Multicast-Kommunikation basierende Auffinden anderer Peers sein. Hybride Peer-to-Peer-Systeme realisieren das Auffinden weiterer Peers hingegen meist unter Zuhilfenahme von verteilt vorgehaltenen Servern und folgen damit bei der Selbstorganisation dem Client-Server-Prinzip. (Steinmetz & Wehrle 2004)

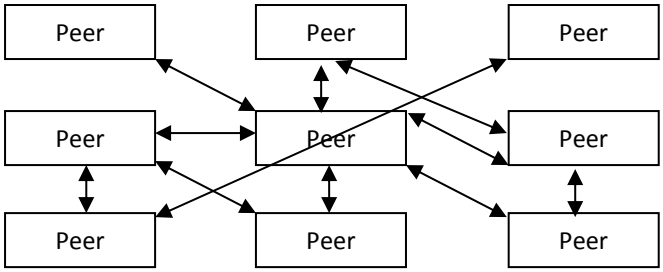



Abbildung 12: Peer-to-Peer-Netz

Abbildung 12 zeigt die Beliebigkeit der Verbindungen und Kommunikationsbeziehungen zwischen der Menge der Knoten in einem Peer-to-Peer-Netzwerk. Die Auffindung von Diensten oder Daten in reinen Peer-to-Peer-Systemen wird häufig mittels Flooding, einem rekursiven Mechanismus zur Befragung aller verfügbaren Peers, realisiert. (Dimakopoulos & Pitoura 2006) Dabei handelt es sich um einen Mechanismus, der bezogen auf den Ausfall einzelner Peers fehlertolerant, aber sehr ressourcenaufwendig ist, da er auf jedem erreichbaren Peer Rechenlast erzeugt.

**Klassifikation anhand der verarbeiteten Daten**

Ein weiterer Ansatz zur Klassifikation medizinischer Informationssysteme ist die Gruppierung der Systeme anhand der von ihnen verarbeiteten Daten. Hier kann für derzeit verbreitete Systeme zwischen den folgenden Gruppen unterschieden werden.

- Systeme der medizinischen Bildverarbeitung und -archivierung
    - Zweidimensionale Bilder
    - Dreidimensionale Bilder
    - Bewegte Bilder oder Filme
- Systeme der textuellen medizinischen Dokumentation
    - Strukturierte Daten
    - Unstrukturierte Daten
- Systeme zur Erfassung und Verwaltung numerischer Messwerte z. B. mit Zeitbezug
    - z. B. EKG-Kurven, Langzeit-EKG oder Kurven der Lungenfunktion

Die Zuordnung zu den angegebenen Gruppen ist nicht exklusiv. Es gibt sowohl Systeme, die sich tatsächlich nur in eine der angegebenen Gruppen einordnen lassen, als auch Systeme, die mehrere oder sogar alle angegebenen Dokumententypen verarbeiten. Außerdem ist die hier angegebene Aufzählung von Typen verarbeiteter Daten nicht vollständig und kann problembezogen erweitert werden.

### 2.2.2. Einordnung verteilter Krankenakten

Verteilte Krankenakten sind anhand der in Kapitel 2.2.1 beschriebenen Klassifikationskriterien als Teilmenge der medizinischen Informationssysteme definierbar.

Die in dieser Arbeit hierfür gewählten Kriterien sind:

- Anwendungsgebiet,
- Verteilung und Gleichartigkeit der Systembestandteile und
- verarbeitete Daten.

Außerdem werden zusätzliche relevante Aspekte berücksichtigt, die nicht bereits in 2.2.1 erläutert sind. Dabei handelt es sich um:

- Verteilungstransparenz und
- Patientenkontext.

Ausgehend von diesen Kriterien lässt sich der Begriff *verteilte Krankenakte* wie folgt in den der *medizinischen Informationssysteme* einordnen:

Eine verteilte Krankenakte ist ein medizinisches Informationssystem, das die in 2.1.2 beschriebene Definition einer elektronischen Krankenakte oder die Definition einer elektronischen Patientenakte aus 2.1.3 erfüllt. Ihr Anwendungsgebiet ist hinsichtlich der in 2.2.1 befindlichen Auflistung beliebig, sofern es medizinischer Natur ist und mindestens den Umfang einer partiellen Krankenakte abdeckt. Das System muss mehr als einen logischen Bestandteil aufweisen und über eine größere Anzahl von Geräten verteilt sein. Außerdem muss das System mehrere der am Ende von Kapitel 2.2.1. aufgelisteten Typen verarbeiteter Daten unterstützen und zur Verfügung stellen. Die aus fachlichen Gründen verarbeiteten Daten haben immer eine Beziehung zu genau einem, über Subsystemgrenzen hinweg eindeutigen Patienten. Ein weiteres notwendiges Kriterium ist die im Vorwort zu Kapitel 2 beschriebene Verteilungstransparenz. Sie ist bezogen auf die von der verteilten Krankenakte angebotenen Schnittstellen ein notwendiges Kriterium für die Zuordnung eines Systems zur Menge der verteilten Krankenakten.

Der vorangegangene Absatz definiert den Begriff *verteilte Krankenakte*. Er beinhaltet eine Menge von Bedingungen, die noch im Detail erläutert und begründet werden müssen.

***Bedingung: elektronische Krankenakte bzw. Patientenakte***

Verteilte Krankenakten sind den elektronischen Kranken- bzw. Patientenakten zuzuordnen, da sie, wie der Name bereits aussagt, Krankenakten sind, die mit Hilfe von Mechanismen verteilter IT-Systeme organisations- oder organisationseinheitsübergreifend die Verwaltung und den Zugriff auf Behandlungs- und Patientendaten ermöglichen.

***Bedingung: freies (medizinisches) Anwendungsgebiet***

Das Anwendungsgebiet verteilter Krankenakten ist innerhalb des Bereichs medizinischer Informationssysteme, bezogen auf die in 2.2.1 gelisteten Kriterien frei. Das heißt, die medizinische Disziplin, in der das System Anwendung findet, ist durch den Begriff *verteilte Krankenakte* nicht definiert und kann eines, mehrere oder alle denkbaren Fachgebiete der Medizin umfassen. Der erforderliche Mindestumfang ist der einer partiellen Krankenakte.

***Bedingung: mehr als ein logischer Bestandteil***

Unter logischen Bestandteilen einer verteilten Krankenakte werden selbstständig und unabhängig verteilbare Software-Elemente wie Software-Komponenten oder Subsysteme verstanden. Die Verteilbarkeit von Software-Elementen setzt voraus, dass deren Anzahl größer oder gleich zwei ist. Deshalb ist für eine verteilte Krankenakte erforderlich, dass sie aus mindestens zwei solchen Elementen besteht. Ein monolithisches System kann folglich keine verteilte Krankenakte sein. Heterogene und komponentenbasierte Systeme sind hingegen grundsätzlich Kandidaten für verteilte Krankenakten.

***Bedingung: über mehrere Geräte verteilt***

Ein System kann, selbst wenn es aus mehreren logischen Software-Bestandteilen besteht, sowohl als ein lokales wie auch ein verteiltes Anwendungssystem realisiert sein. Deshalb muss ein System, um der Menge verteilter Krankenakten zugeordnet zu werden, auf mindestens zwei physikalisch voneinander getrennten, aber durch Netzwerk verbundenen Rechnersystemen verteilt existieren. Aus diesem Grund sind Systeme, die die in 2.2.1 beschriebene Architektur „Host mit Terminals“ realisieren, grundsätzlich keine verteilten Krankenakten. Client-Server-Systeme können erst ab der Verwendung von zwei Datenbanken oder zwei Servern Kandidaten für verteilte Krankenakten sein. Gleiches gilt für mehrschichtige Architekturen. Bei Peer-to-Peer-Systemen muss dazu die Anzahl der Knoten, die eigene Daten für andere Peers zur Verfügung stellen, größer oder gleich zwei sein. Ausschließlich konsumierende Clients oder Peers gelten, bezüglich der Verteilung von Anwendungslogik und Daten, nicht als zusätzliche Systemknoten der Krankenakte.

***Bedingung: mehrere Typen verarbeiteter Daten***

Ein System, das ausschließlich Bilder einer definierten Untersuchung X verarbeitet, gilt nicht als Krankenakte, da eine Krankenakte als Akte eine geordnete Sammlung von zu jemandem gehörigen Dokumenten ist und somit mehrere, auch unterschiedlich geartete Dokumente mit Bezug zu einem Individuum beinhaltet. Außerdem sind ergänzende Angaben zum Verständnis der enthaltenen Informationen notwendiger Bestandteil einer Akte (siehe 2.1.1). Aus diesem Grund wird in der obigen Definition festgelegt, dass eine verteilte Krankenakte mindestens verschiedene Typen textueller Dokumente beinhalten muss und gegebenenfalls zugehörige Bild-, Kurven- oder Video-Dokumente beinhalten kann.

***Bedingung: einheitlicher Patientenkontext***

Um als eine Akte in Erscheinung zu treten, ist es notwendig, dass alle Dokumente aus Subsystemen eindeutig Patienten zugeordnet werden können. Hierzu werden entweder Patienten-Nummern benötigt, die für alle Subsysteme gemeinsam verwendet werden, oder ein Mechanismus, der im Rahmen der Zusammenführung der Dokumente zu einer Akte z. B. durch Mapping eine virtuelle, für die gesamte verteilte Akte gültige eindeutige Patientenzuordnung erreicht.

***Bedingung: Verteilungstransparenz***

Entsprechend der Definition von Tanenbaum und van Steen (Tanenbaum & Van Steen 2003:S. 18) erscheint ein verteiltes System dem Benutzer wie ein einzelnes, kohärentes System. Bei der Betrachtung verteilter Systeme hat sich durchgesetzt, solche Systeme als verteilungstransparent zu bezeichnen. Eine verteilte Krankenakte ist eine Krankenakte, die als verteiltes System konzipiert ist. Aus diesem Grund ist Verteilungstransparenz eine notwendige Bedingung für die Zuordnung eines Systems zur Menge der verteilten Krankenakten. Aber auch der verwendete Begriff der Akte, die eine geordnete Sammlung von zu etwas oder jemanden Bestimmten gehörigen Dokumenten (vgl. 2.1.1) ist, antizipiert ein gemeinsames Ordnungsschema und somit die Notwendigkeit, die Systemarchitektur-bedingte Verteilung der verwendeten Daten vor dem Benutzer oder anderen höher liegenden Schichten zu verbergen. Auch die Forderung nach einem einheitlichen Patientenkontext ist ein Teilaspekt der Verteilungstransparenz.

In der Einleitung zu Kapitel 2 befindet sich die folgende Definition verteilter Krankenakten, die sich im Wortlaut von der in diesem Kapitel erarbeiteten Definition unterscheidet:

„Unter einer verteilten Krankenakte versteht man eine elektronische Kranken- oder Patientenakte, deren Inhalt über mehrere physikalisch oder logisch getrennte Systeme verteilt gespeichert ist, die aber dennoch als eine zusammengehörige Akte in Erscheinung tritt. Es handelt sich dabei folglich nicht um eine beliebige Menge von Systemen, die Daten über Patienten verwalten, sondern um eine Menge von Systemen, die miteinander verbunden sind und einen gemeinsamen Patientenkontext oder einen Mechanismus zur virtuellen Erreichung eines gemeinsamen Patientenkontexts besitzen. Diese Menge von Systemen kann anders formuliert auch eine als kohärentes verteiltes System realisierte Kranken- oder Patientenakte genannt werden.“

Unterschiede in Definitionen haben grundsätzlich negativen Einfluss auf die Schärfe der darauf aufbauend getroffenen Aussagen. Aus diesem Grund ist es erforderlich zu zeigen, dass mindestens die Kernaussagen der beiden Definitionen gleich sind.

Beiden Definitionen liegt die in Abbildung 13 aufgezeigte Eigenschaft verteilter Krankenakten als echte Teilmenge der Menge medizinischer Informationssysteme zugrunde. Somit können medizinische Informationssysteme als Grundgesamtheit betrachtet werden, die durch gezielte Einschränkung ihrer Elemente, anhand deren Eigenschaften, auf die Menge der verteilten Krankenakten reduziert werden kann.

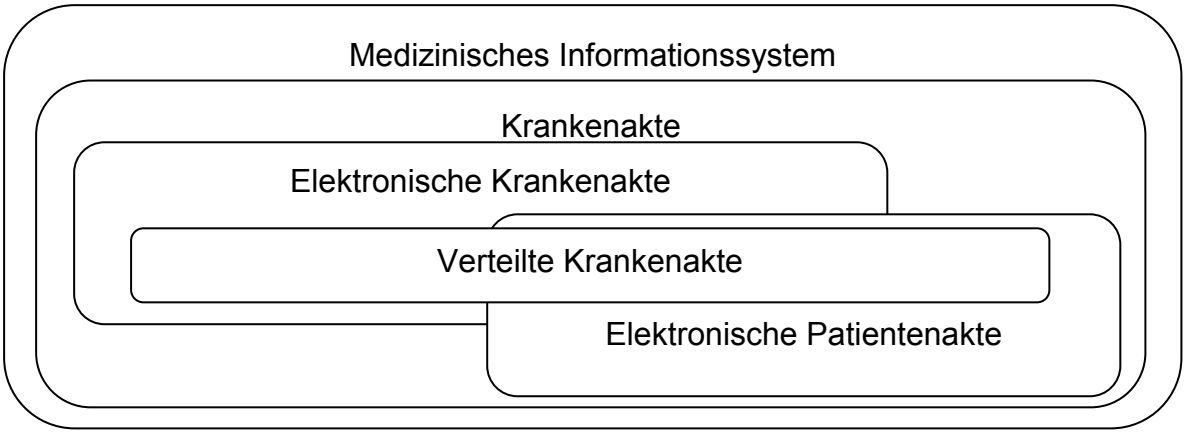



Abbildung 13: Mengendiagramm: Medizinisches Informationssystem

Die in diesem Kapitel (2.2.2) entwickelte Definition wird nachfolgend als Definition A, die in der Einführung zu Kapitel 2 verwendete Definition als Definition B bezeichnet und verwendet.

Definition A besteht aus den folgenden Aussagen:

(1) Ist elektronische Krankenakte oder elektronische Patientenakte
(2) und das Anwendungsgebiet ist innerhalb des medizinischen Bereichs beliebig
(3) und das System besteht aus mehr als einem logischen Bestandteil
(4) und das System ist über mehrere Geräte verteilt
(5) und es verarbeitet unterschiedliche Typen von Daten
(6) und es bietet einen einheitlichen Patientenkontext
(7) und es ist verteilungstransparent

Die Aussage 4 in Definition A, „das System ist über mehrere Geräte verteilt“, ist sehr allgemein und muss weiter konkretisiert werden. Sie bezieht sich nämlich nicht auf die Menge der Clientsysteme, die keine eigene Datenhaltung aufweisen, sondern ausschließlich die Menge der für Serversysteme verwendeten Geräte, die tatsächlich Daten der medizinischen Dokumentation, Bildgebung usw. verwalten. Die Aussage 6 in Definition A zum einheitlichen Patientenkontext umfasst alle technischen Ansätze, die eine eindeutige Identifikation eines realen Patienten über Subsystemgrenzen hinweg ermöglichen.

Die entsprechenden Aussagen der Definition B sind:

(1) Ist elektronische Krankenakte oder elektronische Patientenakte
(2) und der Inhalt ist über mehrere physikalisch und logisch getrennte Systeme verteilt
(3) und tritt als zusammengehörige Akte in Erscheinung
(4) und ist verteiltes System
(5) und (die Subsysteme haben einen gemeinsamen Patientenkontext
(6) oder einen Mechanismus zur virtuellen Erreichung desselben)

Die Aussage 1 ist in beiden Definitionen gleich und enthält implizit für Definition B die Beschränkung des Geltungsbereichs auf den Anwendungsbereich medizinischer Informationssysteme, da elektronische Krankenakten und elektronische Patientenakten echte Teilmengen der Menge medizinischer Informationssysteme (siehe Abbildung 13) sind. In Definition A ist dieser Zusammenhang durch Aussage 2 explizit erwähnt. Somit ist in beiden Definitionen eine eindeutige Einschränkung auf elektronische Krankenakten und elektronische Patientenakten innerhalb der Menge der medizinischen Informationssysteme getroffen.

Des Weiteren enthalten beide Definitionen Aussagen, die die Menge der potenziellen Kandidaten für verteilte Krankenakten anhand ihrer speziellen Eigenschaften als verteilte Systeme einschränken. Definition A trifft hierzu mit ihren Bedingungen 3, 4 und 7 Aussagen zum Grad der Verteilung auf verschiedene Geräte und logische Bestandteile und benennt Verteilungstransparenz als zusätzliches relevantes Kriterium. In Definition B sind die Eigenschaften verteilter Krankenakten als verteilte Systeme in den Aussagen 2 und 4 zusammengefasst. Die Bedingung 4 aus Definition B fordert, dass ein System ein verteiltes System sein muss, um verteilte Krankenakte sein zu können. Wie auf Seite 37 erläutert ist Verteilungstransparenz – ohne nähere Definition von Art und Umfang der Verteilungs-transparenz – eine Eigenschaft verteilter Systeme. Auch Aussage 3 aus Definition B, „tritt als zusammengehörige Akte in Erscheinung“ verweist auf einen speziellen Aspekt der Verteilungstransparenz. Damit enthalten beide Definitionen auch hinsichtlich des Aspekts der Verteilung eine ausreichend große Überdeckung, um hierin als gleich betrachtet werden zu können.

Somit verbleiben in beiden Definitionen noch die Aussagen 5 und 6. Während Aussage 6 in Definition A und Aussage 5 in Definition B äquivalent sind, sind die beiden anderen Aussagen ungleich. Definition B trifft keine expliziten Aussagen zu den Typen verarbeiteter Daten, während Definition A keine Aussage dazu trifft, ob auch ein Mechanismus zur Erreichung eines virtuell gemeinsamen Patientenkontexts gültig ist. Diese Aspekte müssen also, sollen die beiden Definitionen gleich sein, in den verwendeten Begriffen aus deren restlichen Aussagen enthalten sein.

Bezüglich des Mechanismus‘ zur Erreichung eines virtuell gemeinsamen Patientenkontexts kann festgehalten werden, dass der Einsatz eines solchen Mechanismus‘ dazu führt, dass eine verteilte Krankenakte bezüglich der Identifikation einzelner Patienten verteilungstransparent wird, obwohl die einzelnen Subsysteme individuelle Nummernkreise zur Identifikation der Patienten verwenden. Dieser Aspekt ist in den Erläuterungen zur Bedingung „einheitlicher Patientenkontext“ auf Seite 37 als gültiger Fall erwähnt. Somit ist im Wortlaut der Definition A zwar kein expliziter Hinweis auf die Gültigkeit eines solchen Mechanismus enthalten. Er wird allerdings durch die enthaltenen Bedingungen 6 und 7 implizit berücksichtigt.

Lediglich die Bedingung 5 aus Definition A besitzt kein direktes Gegenstück in Definition B. Sie beschreibt mit dem Verweis auf die Notwendigkeit, unterschiedliche Typen von Daten zu verarbeiten, eine Einschränkung, die besonders kleine Systeme ausschließt. Systeme, die in realen Einsatzszenarios als elektronische Krankenakten oder elektronische Patientenakten Verwendung finden, erfüllen diese Bedingung üblicherweise aufgrund der durch sie

abgedeckten Anwendungsfälle, die in der Regel unterschiedliche Typen verarbeiteter Daten erfordern.

Somit ergibt sich mit Ausnahme dieses einen Aspekts eine Gleichheit beider Definitionen in allen ihren zentralen Bestandteilen, wobei Definition A die präzisere der beiden ist. Die ermittelte Gleichheit der Hauptbestandteile beider Definitionen ist die Grundlage für die Gültigkeit der in diesem Kapitel getroffenen Einordnung verteilter Krankenakten in den Kontext medizinischer Informationssysteme. Sie dient als Grundlage für die Bewertung der Relevanz der einzelnen Systeme, die in den Untersuchungen und Aussagen der gesamten vorliegenden Arbeit Verwendung finden.

## 2.3. Anwendungsfälle

Eine eindeutige Definition eines Begriffs dient seiner Verständlichkeit. Der im vorangegangenen Kapitel definierte Begriff *verteilte Krankenakte* beschreibt eine Klasse von Softwaresystemen. Der Zweck von Softwaresystemen ist es, von einer Gruppe dafür vorgesehener Benutzer im Rahmen deren praktischer Tätigkeiten eingesetzt, also genutzt zu werden. Neben der eindeutigen Begriffsdefinition dient deshalb auch die Beschreibung des Einsatzgebiets und der Einsatzszenarios dem Verständnis für die Aufgaben und den Umfang von verteilten Krankenakten.

Use Cases (Anwendungsfälle) werden verwendet, um die notwendige Funktionalität eines Systems mit direktem Bezug zu den beteiligten Akteuren zu beschreiben. In der hier genutzten Form wurden sie erstmals von Ivar Jacobson in „Object-Oriented Software Engineering – A Use Case Driven Approach“ (Jacobson 1997) vorgestellt. Abweichend von der üblichen Verwendung werden Anwendungsfälle in der hier vorliegenden Arbeit nicht für die Beschreibung der Funktionalität eines einzelnen Systems, sondern für die Spezifikation notwendiger und möglicher Funktionalität einer Klasse von Systemen verwendet. Zusätzlich wird als Mittel zur Erhöhung der Präzision der Darstellung auf die Beschreibungsschablone für Use Cases nach Alistair Cockburn (Cockburn 2008:S. 17–18; Cockburn 2000) in einer für die abstraktere Betrachtungsweise einer Klasse von Systemen modifizierten Form zurückgegriffen.

In diesem Kapitel werden die Anwendungsfallbeschreibungen dazu benutzt, die spezifische Funktionalität einer verteilten Krankenakte anwendungsbezogen zu beschreiben, um daraus zu einem späteren Zeitpunkt Kriterien für Erfolg versprechende Softwarearchitekturen abzuleiten. Zudem vertiefen sie das Verständnis für die betrachtete Anwendungsdomäne und erleichtern so die Anwendung der später vorgestellten Muster. Dabei wird explizit der Bezug zwischen der Theorie der zu entwickelnden Pattern-Sprache (Pattern Language) und deren praktischem Anwendungsgebiet hergestellt, was besonders im Hinblick auf die Tatsache, dass sich die Medizininformatik in den Bereich der angewandten Informatik-Disziplinen einordnet, eine wesentliche Voraussetzung für die Begründung der Relevanz dieser Arbeit ist. Außerdem dient die Beschreibung der Anwendungsfälle auch dazu, den Wirkungsbereich der zu entwickelnden Pattern-Sprache einzugrenzen.

### 2.3.1. Ausgewählte Anwendungsfälle

Verteilte Krankenakten subsumieren den Verteilungsaspekt sowohl von elektronischen Krankenakten als auch von elektronischen Patientenakten. Aus diesem Grund werden hier zur Darstellung sowohl Anwendungsfälle mit Bezug zu elektronischen Krankenakten als auch solche mit Bezug zu elektronischen Patientenakten gewählt. Diese bilden dann die Basis für die Findung des abstrakten Anwendungsfalls für verteilte Krankenakten der in Kapitel 2.3.2. vorgestellt wird.

#### 2.3.1.1. Gemeinsame Akteure

Akteure bilden in Anwendungsfalldiagrammen je eine Menge von Systemteilnehmern ab, die ein System in einer bestimmten Rolle benutzen. Jacobson definiert den Zusammenhang zwischen realen Personen und Akteuren wie folgt:

*„One person can instantiate (play the roles of) several different actors. Actors thus define roles that users can play“* (Jacobson 1997:S. 157).

Um die in den nachfolgenden Abschnitten beschriebenen Anwendungsfälle vergleichbar zu gestalten, wird auf eine Menge von sechs abstrakten Akteuren zurückgegriffen. Diese sind durch Abstraktion der Inhalte von vier Quellen entwickelt, die die Aufgaben einer elektronischen Krankenakte aus verschiedenen Blickwinkeln beschreiben. Der Bereich der inhaltlichen Zusammenhänge und Gestaltungsformen wird primär durch die ausführliche Beschreibung der Module einer elektronischen Krankenakte durch Haas (Haas 2009:S. 275–419) abgedeckt. Der Bereich der sicherheits- und rollenbezogenen Betrachtung stützt sich auf die von Ochsenschläger u. a. beschriebenen Rollen zur Sicherheitsstrategie einer elektronischen Krankenakte (Ochsenschläger u. a. 2008) sowie die ebenfalls von Haas beschriebenen Akteure im Gesundheitsweisen (Haas 2006:S. 179ff.). Die vierte der verwendeten Quellen ist eine Sammlung von Use-Case-Beschreibungen (HEAL NY Phase 5 Health IT & Public Health Team 2008) des New Yorker Department of Health.

Ein wesentliches Kriterium bei der Auswahl dieser Akteure ist, dass sie unabhängig von beruflichen Abschlüssen und Qualifikationen eine Menge von logisch zusammengehörigen Aufgaben und Eigenschaften zusammenzufassen. So wird beispielsweise bei der Erbringung und Dokumentation von Leistungen nicht zwischen Ärzten und Pflegepersonal unterschieden, da deren Tätigkeiten bezogen auf das System der elektronischen Krankenakte gleich sind. Deshalb verbessert sich durch die Verwendung der Akteure die Verständlichkeit der Prozessdiagramme. Allerdings ist dazu eine klar verständliche und eindeutige Bezeichnung des Akteurs notwendig.

Zusätzlich wird in der Beschreibung der sechs abstrakten Akteure auch die Wirkungsweise der zentralen Aktionen ihrer Tätigkeiten, basierend auf den grundlegenden CRUD-Operationen (Kilov 1990) (Create, Read, Update, Delete), beschrieben um das Verständnis für deren Art der Interaktion mit dem System zu verbessern.

**Akteur: Patient**
Der Akteur Patient ist bezogen auf das Anwendungssystem passiv. An ihm werden Leistungen (Behandlungsschritte und Untersuchungen) erbracht und dokumentiert. Seine Stammdaten werden erfasst. Er ist der initiale Auftraggeber für einen Behandlungsprozess oder Behandlungspfad. In derzeitigen elektronischen Krankenakten führt er keinerlei Aktionen aus.

In zukünftigen Patienten- oder Gesundheitsakten sollen allerdings auch Patienten aktiv am System teilnehmen können (Riebling 2007). Eine der möglichen künftigen Aufgaben ist

hierbei die Vergabe von Zugriffsberechtigungen auf medizinische Dokumente zur Wahrung seines Rechts auf informationelle Selbstbestimmung.

**Akteur: Leistungserbringer**
Der Akteur Leistungserbringer ist für die Durchführung oder Erbringung medizinischer Leistungen zuständig. Medizinische Leistungen sind dabei die Untersuchung, Diagnostik, Behandlung und Pflege. Der Leistungserbringer erhebt als untersuchender Mitarbeiter mit oder ohne technische Hilfsmittel den Zustand des Patienten. Dabei kann die Palette der Hilfsmittel von einfachen Mitteln der körperlichen Untersuchung bis hin zu komplexer Geräte- und Labormedizin reichen. Als behandelnder Mitarbeiter führt der Leistungserbringer therapeutische oder pflegerische Aktionen mit dem Ziel der Heilung oder Linderung am Patienten durch.

Bezogen auf das System einer elektronischen Krankenakte werden Leistungserbringer vorwiegend lesend tätig. Das System bietet Ihnen die Möglichkeit, sich über die medizinische Vorgeschichte, aktuelle Diagnosen und sonstige wichtige Eigenschaften des Patienten zu informieren. Diese Informationen sind die Voraussetzung, um die angeforderte Leistung korrekt erbringen zu können.

**Akteur: Auftraggeber**
Die Erbringung einer medizinischen Leistung wird meist durch die Erteilung eines klinischen oder externen Auftrags initiiert. Der Akteur Auftraggeber erteilt diesen Auftrag und beantragt somit medizinische Leistungen bei anderen Leistungsstellen. Im Fall einer elektronischen Patientenakte, die per Definition einrichtungsübergreifend ist, unterscheidet man zwischen internem und externem Auftraggeber.

Der Auftraggeber einer Leistung ist häufig der Erbringer einer vorangegangenen Leistung. Die Erbringung einer diagnostischen Leistung führt beispielsweise meist entweder zu einer weiterführenden diagnostischen oder einer daraus resultierenden therapeutischen Leistung.

Der Akteur Auftraggeber wird im System schwerpunktmäßig schreibend tätig. Lesende Tätigkeiten sind kein inhärenter Bestandteil seiner Aufgabe, können allerdings aus fachlicher Sicht für deren korrekte Durchführung notwendig sein.

**Akteur: Dokumentar**
Die Aufgabe des Akteurs Dokumentar ist es, Erkenntnisse aus Diagnostik, Behandlung und Pflege sowie durchgeführte Leistungen so zu erfassen, dass dabei eine lückenlos nachvollziehbare Krankengeschichte entsteht. In der Praxis sind Dokumentar und Leistungserbringer oft dieselbe natürliche Person.

Bezogen auf das System einer elektronischen Krankenakte ist die Erstellung von Dokumenten die Aufgabe des Dokumentars. Er ist folglich hauptsächlich schreibend tätig.

Die Rolle des Dokumentars wird üblicherweise von den Berufsgruppen Arzt, Pflegekraft oder medizinischer Dokumentar ausgeführt. Bei den in der Hierarchie höher stehenden Ärzten ist die Aufgabe der Dokumentation meist auf den Arzt selbst und dessen Schreibkraft verteilt.

**Akteur: Verwalter**
Dem Akteur Verwalter obliegt die Aufnahme von Patienten, die Verwaltung von deren Stammdaten und die Abrechnung medizinischer Leistungen. Durch die Aufnahme von Patienten steht er, abhängig von den vorhandenen Ausgangsinformationen, am Beginn eines jeden Behandlungspfads. Wegen seiner Aufgabe, die erbrachten Leistungen abzurechnen, steht er auch am Ende desselben. Der Verwalter ist für einen Großteil der nicht medizinischen Aufgaben, die durch die Behandlung von Patienten anfallen, zuständig.

Im System einer elektronischen Krankenakte ist der Verwalter sowohl lesend, schreibend als auch aktualisierend tätig. Seine Tätigkeit erweitert den Umfang der Akte unter anderem um nicht medizinische Informationen über den Patienten.

**Akteur: medizinisches Gerät**
Der Akteur medizinisches Gerät kann ausschließlich durch technische Systeme und nicht durch natürliche Personen repräsentiert werden. Medizinische Geräte unterscheiden sich vom Leistungserbringer, da sie über die Daten des Auftrags hinaus nicht lesend auf existierende Daten zum Patienten zugreifen, sondern stattdessen während ihrer Verwendung automatisch Daten über den Patienten erheben und speichern. Somit führen medizinische Geräte automatisch einen Teil der Dokumentation durch, der vom Leistungserbringer zusätzlich bewertet und vom Dokumentar in die von ihm erstellte umfassende Falldokumentation integriert wird.

Ein praktisches Beispiel hierfür ist die Durchführung einer Röntgenuntersuchung mittels eines Computertomografen (CT). Hierbei speichert das CT automatisch eine Menge von Bildern in einem Bildarchiv. Die Erfassung textueller Ergebnisdaten wird von einem Akteur der Rolle Dokumentar durchgeführt.

Bezogen auf das System einer elektronischen Krankenakte wird der Akteur medizinisches Gerät überwiegend schreibend aktiv.

#### *2.3.1.2. Anwendungsfälle zur Krankenakte*

Eine Krankenakte ist als geordnete Sammlung von Dokumenten über den Zustand eines Patienten, die innerhalb einer Institution verwendet wird, definiert. In ihr werden Patienten- und Falldaten verwaltet sowie Ergebnisse dokumentiert. Weitere relevante und enthaltene Bereiche der Dokumentation sind die Pflege, Diagnostik, Behandlung und Medikation sowie die Ergebnisse von Laboruntersuchungen und klinische Notizen (Haas 2009:S. 275 ff.). Sie ist ein Hilfsmittel zur Dokumentation verschiedener klinischer Aktivitäten.

Klinische Abläufe, also die kontextabhängige Aneinanderreihung verschiedener Aktivitäten, werden in Form von sogenannten klinischen Pfaden (vgl. Kahla-Witzsch & Geisinger 2004; Hellmann 2003; Böckmann & Houta 2008) bezogen auf festgelegte auslösende Ereignisse definiert und dokumentiert. Klinische Pfade oder Behandlungspfade sind somit die aus medizinischer Sicht abgebildeten Geschäftsprozesse einer Versorgungseinrichtung.

Das Queensland Health Clinical Pathways Board definiert den Begriff des *Clinical Pathways*, der als englischsprachiges Synonym für den Begriff *Behandlungspfad* gilt wie folgt:

*"Clinical pathways are standardized, evidence-based multidisciplinary management plans, which identify an appropriate sequence of clinical interventions, timeframes, milestones and expected outcomes for a homogenous patient group."* (Queensland Health Clinical Pathways Board 2002)

Klinische Pfade können als Ausgangspunkt bei der Erstellung eines Prozessmodells für die IT-Unterstützung von Behandlungsprozessen zu bestimmten Ausgangsindikationen verwendet werden. Konkrete Prozess-Szenarios können dabei als Anwendungsfälle beschrieben und betrachtet werden. Die Untersuchung von Anwendungsfällen dient hierbei der Betrachtung und Beschreibung verschiedener Szenarios der Nutzung von bestehenden oder zukünftigen Softwaresystemen. Oesterreich nennt deshalb auch den Begriff *Nutzungsfalldiagramm* als mögliches deutschsprachiges Synonym zu *Anwendungsfalldiagramm* (Oestereich 2009:S. 245).

Als Grundlage für die Erläuterung der Funktionsweise elektronischer Kranken- resp. elektronischer Patientenakten werden die folgenden beiden Anwendungsfälle verwendet:

- Patient mit Herzinsuffizienz an Notaufnahme und
- Patient mit ambulant erworbener Lungenentzündung.

Beiden Anwendungsfallbeschreibungen liegen veröffentlichte klinische Pfade deutscher Universitätskliniken zugrunde. Der Anwendungsfall zu Herzbeschwerden basiert auf dem Behandlungspfad Herzinsuffizienz aus der medizinischen Klinik II des Universitätsklinikums Regensburg (Bergler 2008:Folie 15). Für den Anwendungsfall Lungenentzündung wird der „Clinical Pathway Ambulant erworbene Pneumonie" des Universitätsklinikums Aachen (Krüger 2008) verwendet. Initiator der Behandlung ist in beiden Anwendungsfällen der Akteur Patient, der durch sein Erscheinen bei der Versorgungseinrichtung das auslösende Ereignis liefert.

**Patient mit Herzinsuffizienz an Notaufnahme**

Wie bereits erwähnt, wird für die textuelle Beschreibung der Use Cases eine an die Problemstellung angepasste Form der Beschreibungsschablone nach Cockburn (Cockburn 2008:S. 17–18) verwendet. Der Behandlungspfad Herzinsuffizienz beinhaltet sowohl Patienten mit schwerer Dekompensation als auch solche mit leichter bis mittlerer Dekompensation. Der hier beschriebene Anwendungsfall schließt aus Gründen der Übersichtlichkeit Patienten mit schwerer Dekompensation aus.

Anwendungsfall: Versorgung eines Patienten mit Herzinsuffizienz ab Notaufnahme

Beteiligte Akteure:

- Patient
- Leistungserbringer
- Dokumentar
- Auftraggeber
- Verwalter
- Medizinisches Gerät *(Ruhe-EKG, Röntgenanlage, Echokardiografie-Gerät)*

Auslöser: Ein Patient mit Symptomen einer Herzinsuffizienz wird durch einen Rettungswagen oder Angehörige an der Notaufnahme eines Krankenhauses eingeliefert.

Vorbedingungen: Es liegt keine schwere Dekompensation vor.

Beschreibung: Nach seiner Aufnahme wird der Patient, sofern möglich, über seine bisherige Anamnese befragt und einer ersten Untersuchung unterzogen. Dabei werden seine Stammdaten, sofern noch nicht im System vorhanden, von einem verwaltenden Akteur erfasst. Anschließend wird eine Eingangsuntersuchung am Patienten durchgeführt und deren Ergebnisse im System dokumentiert. Nach dieser ersten Untersuchung entscheidet der verantwortliche Leistungserbringer ob eine schwere Dekompensation vorliegt und gibt abhängig davon unterschiedliche weitere Untersuchungen in Auftrag.

Liegt keine schwere Dekompensation vor, werden eine Laboruntersuchung, ein Ruhe-EKG, eine Röntgen-Thorax-Aufnahme und eine Echokardiografie angeordnet. Diese Untersuchungen werden jeweils von dritten Leistungserbringern durchgeführt und von diesen selbst oder von deren Gehilfen dokumentiert. Die verwendeten medizinischen Geräte dokumentieren dabei die ermittelten Parameter der Untersuchung automatisch. Bei der Dokumentation der Untersuchungsergebnisse werden die von den Medizingeräten ermittelten Parameter, Kurven und Filme vom Dokumentar berücksichtigt und in die entstandenen Dokumente eingearbeitet.

Falls notwendig, erfolgt vor oder während den Untersuchungen eine $O_2$-Insufflation (dem Patienten wird zusätzlicher Sauerstoff zugeführt), wobei deren Beginn- und Endzeitpunkt ebenfalls dokumentiert wird. Sollten keine anderen Diagnosen ermittelt worden sein, so folgt die stationäre Aufnahme auf einer Normalstation, die durch einen Verlegungsauftrag initiiert wird. Der Verlegungsauftrag wird vom bisher verantwortlichen Leistungserbringer gestellt und von verwaltenden Akteuren weiterbearbeitet.

Nachbedingungen: Nach dem erfolgreichen Durchlauf dieses Anwendungsfalls ist der betroffene Patient aufgenommen und untersucht. Die erste Diagnose einer kardialen Dekompensation wurde bestätigt und der Patient zur Weiterversorgung auf eine Normalstation verlegt. Parallel dazu wurden verschiedene elektronische Befund-Dokumente erstellt und einer systematisch geführten Krankenakte hinzugefügt.

Exemplarischer Ablauf in Aktivitätsdiagrammform:

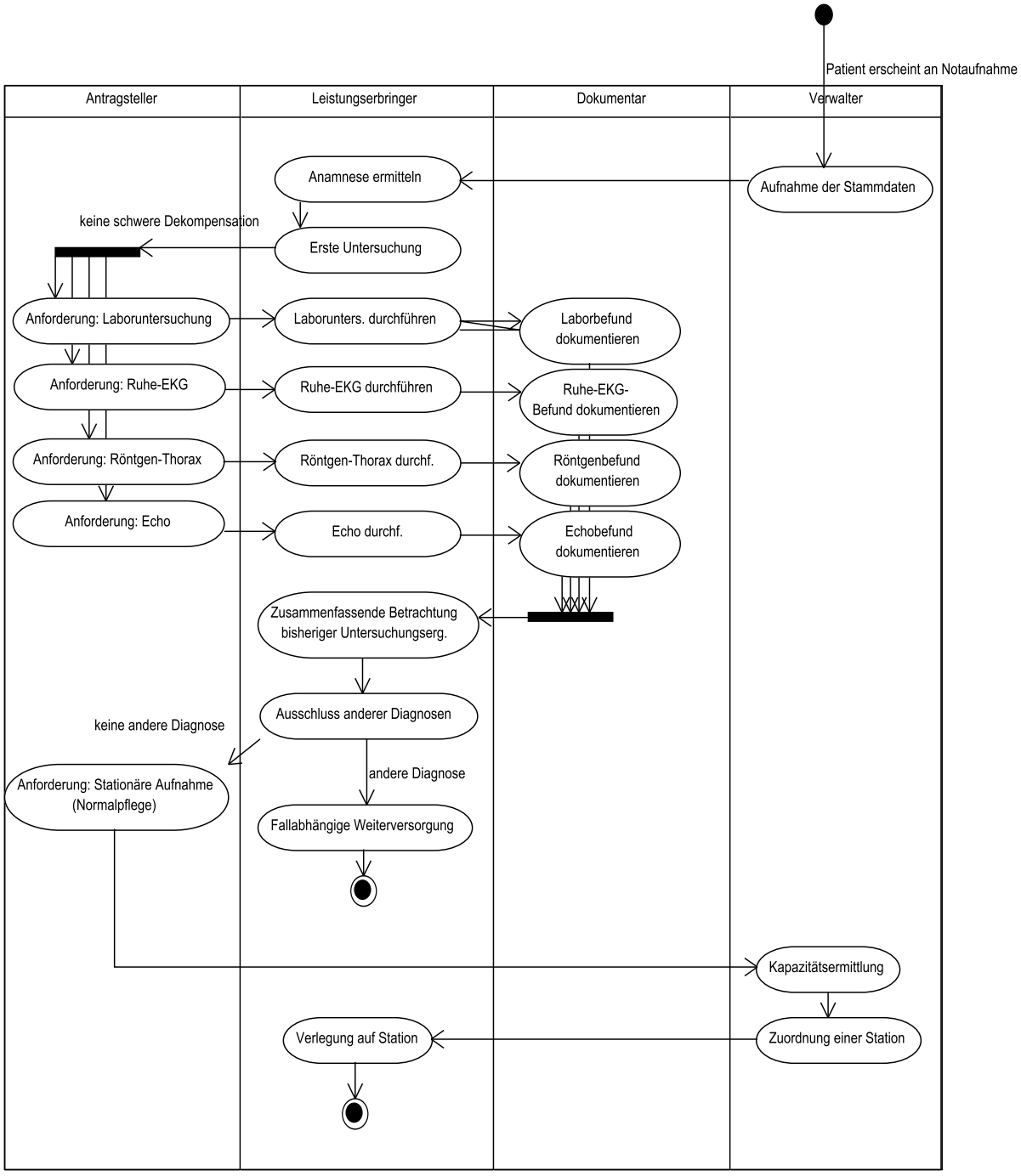



Abbildung 14: Aktivitätsdiagramm zum Anwendungsfall Patient mit Herzinsuffizienz

Hinweise:

Aktivitätsdiagramm zum vollständigen Behandlungspfad Herzinsuffizienz des UK.R:

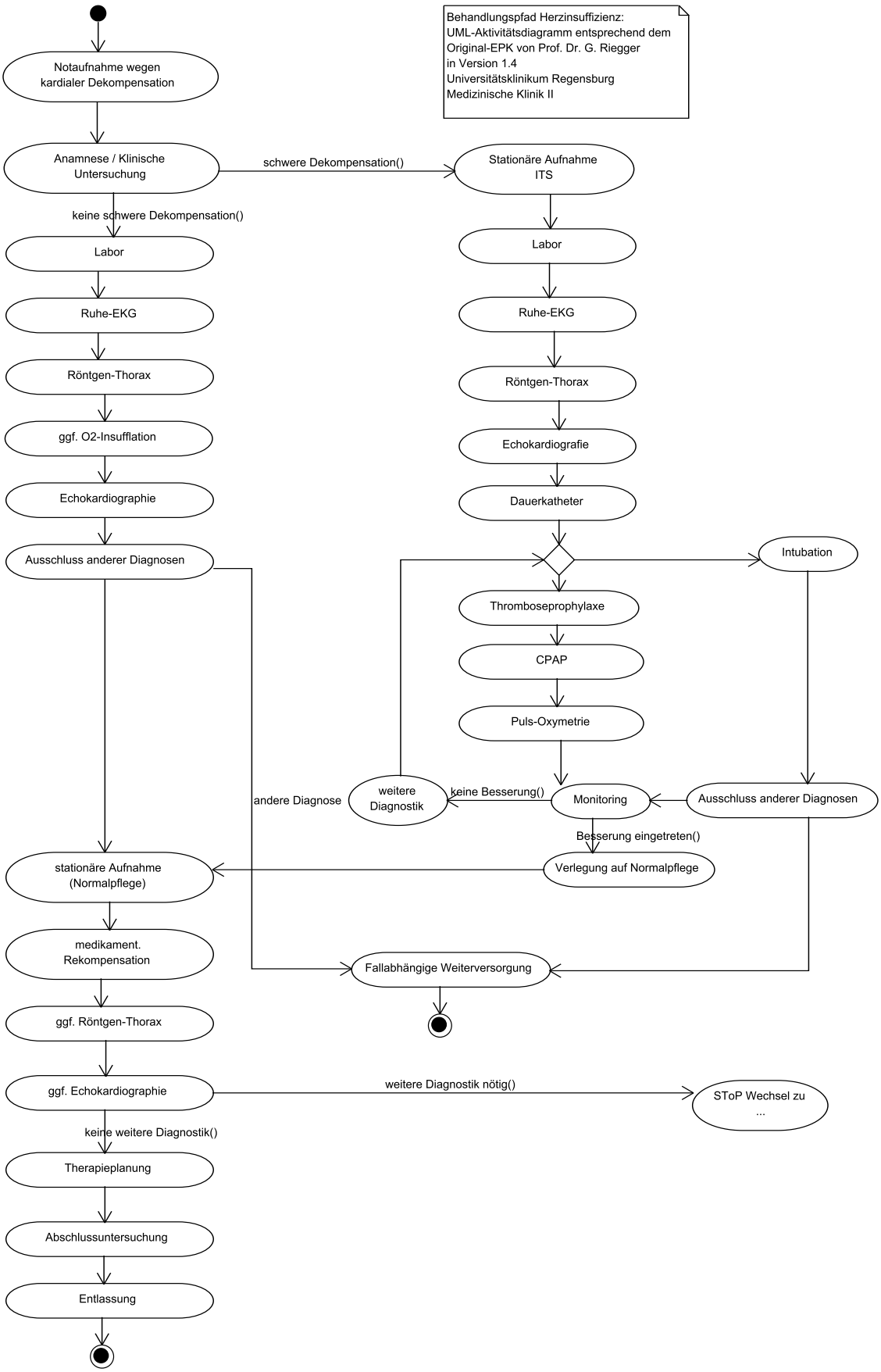



Abbildung 15: Behandlungspfad Herzinsuffizienz am Universitätsklinikum Regensburg (vollständig)

**Patient mit ambulant erworbener Lungenentzündung**

Anwendungsfall: Patient mit ambulant erworbener Lungenentzündung

Beteiligte Akteure:

- Patient
- Leistungserbringer
- Dokumentar
- Auftraggeber
- Verwalter
- Medizinisches Gerät

Auslöser: Ein Patient mit Symptomen einer Lungenentzündung oder einer anderen schweren bronchialen Erkrankung erscheint in der Notaufnahme eines Krankenhauses.

Vorbedingungen: Der Patient hat eine schwere Lungenentzündung. Die Kriterien für die Durchführung eines Röntgen-Thorax sind erfüllt.

Beschreibung: Zuerst werden Daten zum Patienten (Stammdaten u. Ä.) durch einen Akteur der Rolle Verwalter und Daten bezüglich der Anamnese des Patienten durch einen Leistungserbringer erfragt und erfasst. Anschließend wird durch den Akteur der Rolle Leistungserbringer eine Eingangsuntersuchung durchgeführt. Diese sichert gegebenenfalls bei der ersten Befragung ermittelte Informationen ab oder vervollständigt das Bild des Leistungserbringers über die aktuelle Situation des Patienten. Die Ergebnisse der Eingangsuntersuchung werden während ihrer Durchführung durch eine dritte Person der Rolle Dokumentar dokumentiert.

Danach wird durch einen Leistungserbringer geprüft, ob der Zustand des Patienten die Kriterien für die Anordnung einer Röntgen-Thorax-Aufnahme erfüllt. Sind diese Kriterien erfüllt, folgt die Anmeldung des Patienten zur Durchführung einer Röntgenaufnahme des Thorax. Die Rolle des Antragsstellers nimmt hierbei die gleiche Person ein, die vorher in der Rolle des Leistungserbringers die für die Durchführung der Untersuchung notwendigen Kriterien geprüft hat. Anschließend führt ein Leistungserbringer der radiologischen Abteilung die Röntgen-Thorax-Aufnahme durch und befundet sie. Die Röntgenanlage ordnet, basierend auf dem klinischen Auftrag, die Aufnahme automatisch dem richtigen Patienten zu und unterstützt so den Akteur Dokumentar bei der Dokumentation des Befunds.

Der primäre Leistungserbringer erhält das entstandene Befunddokument und entscheidet auf dessen Grundlage über die weitere Versorgung des Patienten. Zeigt der Befund, dass pneumonisches Filtrat vorhanden ist, so ist eine Risikostatifizierung durchzuführen und zu dokumentieren. Dieser Anwendungsfall behandelt das Vorliegen einer besonders schweren Pneumonie und führt folglich, parallel zur Vervollständigung der Dokumentation, zu einer Verlegung auf eine Intensivstation. Diese wird vom primären Leistungserbringer in der Rolle des Antragstellers durch die Anlage eines klinischen Auftrags bezüglich der Verlegung des

Patienten auf eine Intensivstation angestoßen und von einem Akteur der Rolle Verwalter bearbeitet. Anschließend wird der Patient von einem Akteur Leistungserbringer auf die zugewiesene Intensivstation gebracht.

Nachbedingungen: Nach dem erfolgreichen Durchlauf dieses Anwendungsfalls ist der betroffene Patient aufgenommen und untersucht. Die erste Diagnose einer schweren Pneumonie ist bestätigt und der Patient zur Weiterversorgung auf eine Intensivstation verlegt. Dabei sind verschiedene elektronische Befund-Dokumente erstellt und einer systematisch geführten Krankenakte hinzugefügt worden.

Exemplarischer Ablauf in Aktivitätsdiagrammform:

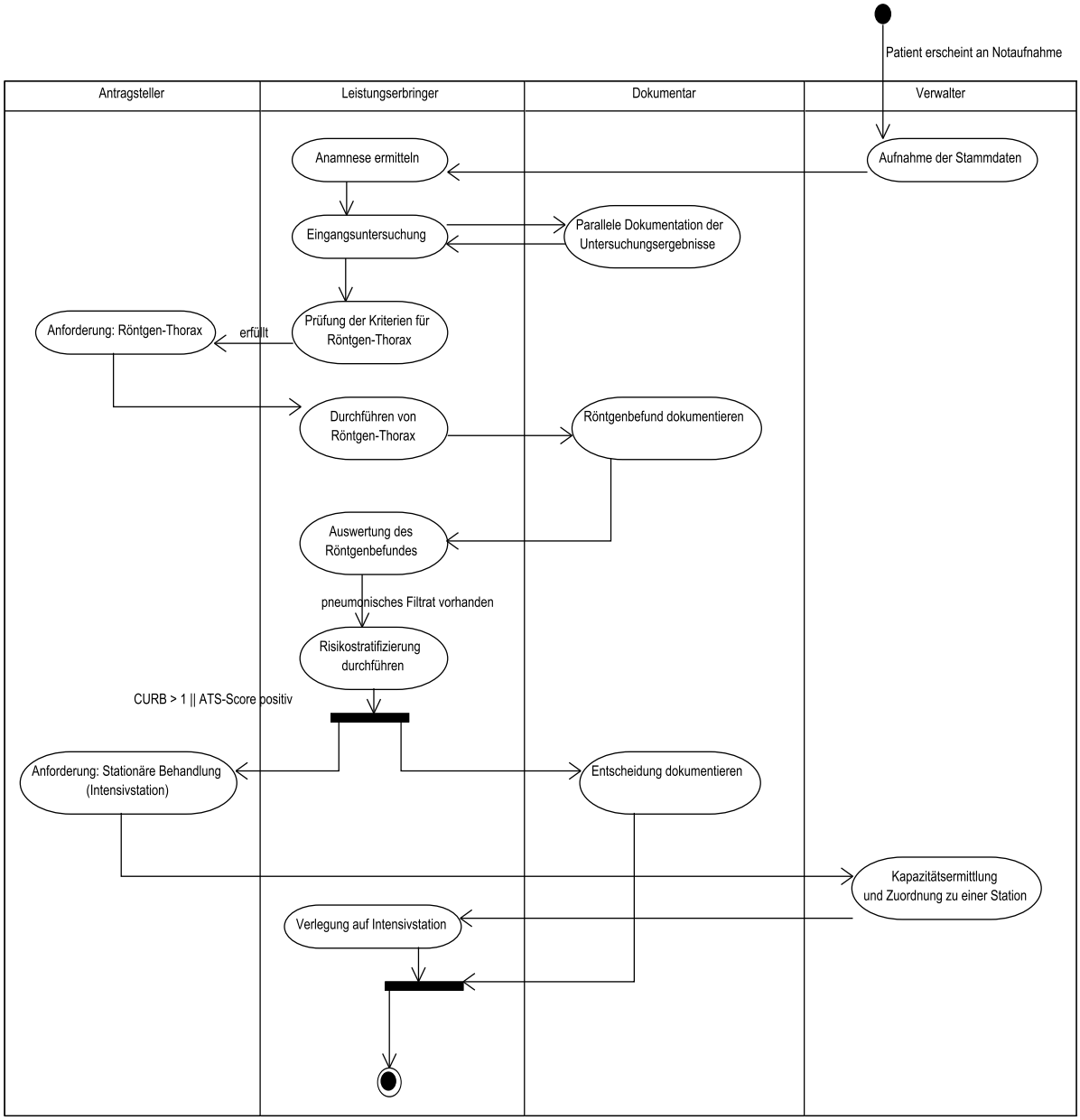



Abbildung 16: Aktivitätsdiagramm zum Anwendungsfall Patient mit schwerer ambulant erworbener Pneumonie

Hinweise: Klinischer Pfad „Ambulant erworbene Pneumonie“ (CAP) des Universitätsklinikums Aachen (Krüger 2008):

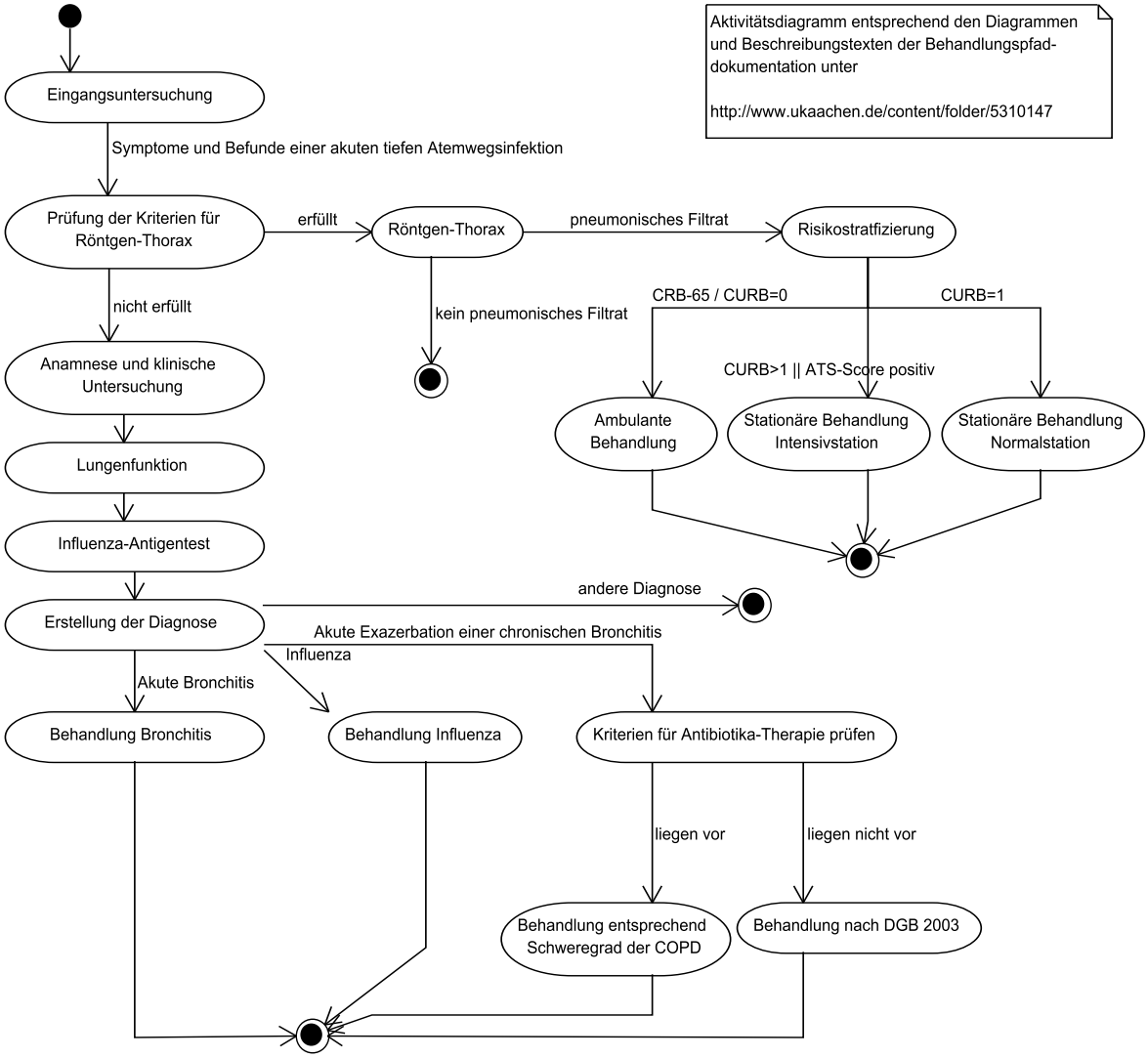



Abbildung 17: Aktivitätsdiagramm zum klinischen Pfad Ambulant erworbene Pneumonie

### *2.3.1.3. Anwendungsfall zur Patientenakte*

Eine Patientenakte ist bezogen auf die Menge der beteiligten Institutionen umfassender ausgerichtet als eine Krankenakte. Während eine Krankenakte sich auf die Leistungsstellen einer einzelnen Institution (z. B. eines einzelnen Krankenhauses) beschränkt, zielt der Ansatz der Patientenakte auf eine institutionsübergreifende Verwendung der Akte.

Die Anwendungsfälle der Kranken- und der Patientenakte unterscheiden sich lediglich darin, dass die Anwendungsfälle der Patientenakte grundsätzlich auch den Übergang von einer Institution zur nächsten berücksichtigen. In derzeitigen medizinischen Informationssystemen ist dieser Übergang von einer Institution zu einer anderen das wesentliche technische Problem, das eine institutionsübergreifende Nutzung der Patientendaten behindert (Blobel 2007:S. 2). Mit der Realisierung elektronischer Patientenakten soll die Idee vollständig technisch unterstützter einrichtungsübergreifender Behandlungspfade praktisch umgesetzt werden. Ein praktisches Beispiel für den Versuch der Umsetzung solcher einrichtungsüber-

greifender oder transsektoraler Pfade ist das Projekt DiPP (Eckenbach & Böckmann 2008) der Knappschaft-Bahn-See.

Als exemplarischer Anwendungsfall einer elektronischen Patientenakte wird der Anwendungsfall „Herzinsuffizienz“ aus dem Kapitel 2.3.1.2 um den stationären Aufenthalt, eine darauf folgende Reha-Maßnahme und die anschließende Weiterversorgung beim Hausarzt erweitert.

**Herzinsuffizienz mit anschließender Weiterbehandlung und Rehabilitation**

Anwendungsfall: Behandlung schwerer Herzinsuffizienz ab Notaufnahme, mit stationärer Behandlung, Rehabilitationsmaßnahme in einer Reha-Klinik und anschließender Weiterversorgung durch den Hausarzt.

Beteiligte Akteure:

- Patient
- Leistungserbringer
  *(Notaufnahme, Normalstation, Leistungsstelle, Hausarzt, Reha-Klinik)*
- Dokumentar
  *(Notaufnahme, Normalstation, Leistungsstelle, Hausarzt, Reha-Klinik)*
- Auftraggeber
  *(intern, extern)*
- Verwalter
  *(Krankenhaus, Hausarzt, Reha-Klinik)*
- Medizinisches Gerät

Beteiligte Institutionen:

- Krankenhaus
  - Notaufnahme
  - Normalstation
  - sonstige Leistungsstellen
- Hausarzt
- Reha-Klinik

Auslöser: Ein Patient mit Symptomen einer Herzinsuffizienz wird durch einen Rettungswagen oder durch Angehörige in die Notaufnahme eines Krankenhauses eingeliefert.

Vorbedingungen:

- Es liegt keine schwere Dekompensation vor.
- Der Patient hat einen Hausarzt.
- Die Form der Erkrankung macht eine nachfolgende Reha-Maßnahme notwendig.
- Die Reha-Maßnahme kann direkt im Anschluss an die stationäre Behandlung im Krankenhaus angetreten werden.

Beschreibung: Nach der Ankunft des Patienten in der Notaufnahme des Krankenhauses wird entsprechend dem Anwendungsfall „Patient mit Herzinsuffizienz an Notaufnahme“ verfahren. Parallel dazu werden zusätzliche administrative Aufgaben ausgeführt, die durch die institutionsübergreifende Nutzung der Dokumentation in der elektronischen Patientenakte (ePA) erforderlich werden. Dabei handelt es sich beispielsweise um die Erfassung des Hausarztes und ggf. des einweisenden Arztes sowie die Einwilligung des Patienten zur elektronischen Weitergabe von Befunden. Anschließend an die im Anwendungsfall „Patient mit Herzinsuffizienz an Notaufnahme“ beschriebene Aufnahme und eingehende Untersuchung wird der Patient solange stationär versorgt, bis ein definierter Genesungszustand erreicht ist, ab dem eine weitere stationäre Versorgung des Patienten nicht mehr notwendig ist. Daraufhin wird ein Arztbrief verfasst und ggf. die noch nicht abgeschlossene Dokumentation vervollständigt. Ist das geschehen, kann der fallbezogene Inhalt der lokalen Krankenakte an die Reha-Klinik, den Hausarzt oder eine zentrale Patientenakte übermittelt, beziehungsweise zum direkten Abruf für die Reha-Klinik und den Hausarzt bereitgestellt werden.

Im Anschluss an die medizinische Versorgung des Patienten in einem Krankenhaus wird als nächster Abschnitt des institutionsübergreifenden Versorgungsprozesses eine stationäre Rehabilitationsmaßnahme an einer Reha-Klinik durchgeführt. Ziel der Maßnahme ist es, die Folgebeschwerden zu lindern. Dabei wird der Gesundheitszustand des Patienten durch gezielt therapeutisch eingesetzte, zumeist sportliche Übungen schrittweise verbessert. Auch Reha-Maßnahmen werden zunehmend in Form von Behandlungspfaden dokumentiert[3]. Nach erfolgreicher Beendigung der Reha-Maßnahme wird vom dort behandelnden Arzt ein Arztbrief als Entlassungsdokument verfasst und der Patient zurück an seinen Hausarzt überwiesen. Zugleich werden die im Laufe der Rehabilitationsmaßnahme entstandene Dokumentation und der Arztbrief an den Hausarzt oder in eine zentrale Patientenakte übermittelt, beziehungsweise für den direkten Abruf durch den Hausarzt und andere berechtigte Leistungserbringer bereitgestellt.

Das Eintreffen des Patienten beim Hausarzt ist der Startpunkt für den dritten Abschnitt des integrierten Behandlungsprozesses. Zur Weiterversorgung muss sich der Arzt als Leistungserbringer einen Überblick über die Anamnese und den aktuellen Zustand des Patienten verschaffen. Dazu liest er in der elektronischen Patientenakte die für den Behandlungsfall relevanten Befund-Dokumente sowie die Arztbriefe aus dem Krankenhaus und der Reha-Klinik. Anschließend folgt er seinem lokalen Behandlungspfad. Dabei führt er, um sicherzustellen, dass sich keine Veränderung des dokumentierten Zustands eingestellt hat, noch einige eigene Untersuchungen am Patienten durch. Die Ergebnisse dieser Untersuchungen werden ebenfalls in der elektronischen Patientenakte dokumentiert. Abhängig vom Zustand des Patienten verordnet der Leistungserbringer dem Patienten Medikamente und legt mit dem Patienten ein Wiederholungsintervall fest, in dem der Patient regelmäßig zur Überprüfung des Gesundheitszustands beim Hausarzt zu erscheinen hat.

Nachbedingungen: Die Behandlung im Krankenhaus, die nachfolgende Reha-Maßnahme und die fallbezogene Weiterbehandlung durch den Hausarzt sind abgeschlossen. Der Patient

[3] Beispiel: Foliensatz zu Vortrag über Behandlungspfade einer Schweizer Reha-Klinik: http://www.igptr.ch/cms/uploads/PDF/PTR/archiv/ptrbehandlungspfade.pdf (insb. S. 14 und 27)

ist genesen und hat den Versorgungsprozess verlassen. Der gesamte Versorgungsprozess ist zusammenhängend dokumentiert. Die entstandene Dokumentation kann später im Bedarfsfall durch einen dazu autorisierten Leistungserbringer in ihrem gesamten Umfang eingesehen und für die Bewertung einer Folgeerkrankung verwendet werden.

Exemplarischer Ablauf in Aktivitätsdiagrammform:

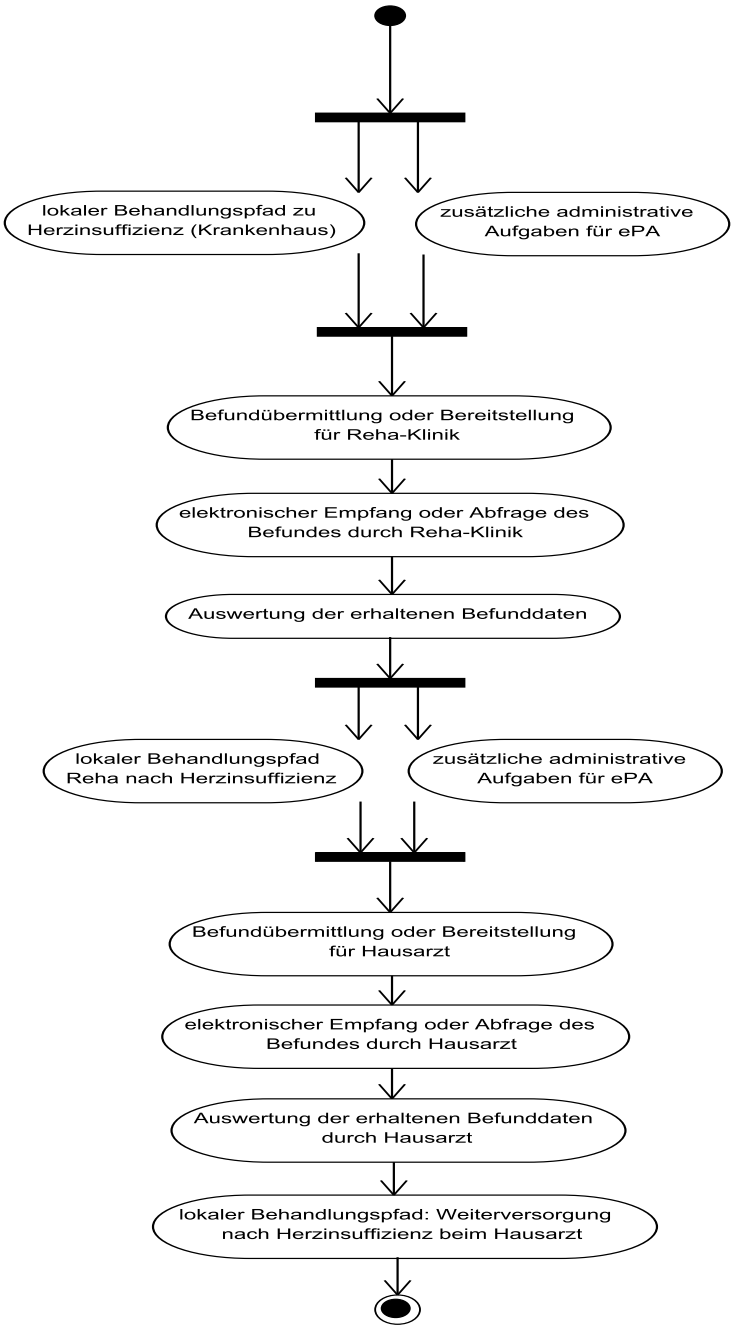



Abbildung 18: Aktivitätsdiagramm zum Anwendungsfall elektronische Patientenakte

### 2.3.2. Abstrakter Anwendungsfall

Die in Kapitel 2.3.1 dargestellten Anwendungsfälle haben Gemeinsamkeiten. Aus ihnen lässt sich eine gemeinsame allgemeinere Darstellungsform erarbeiten. Dazu werden Eigenschaften und Muster, die in allen beschriebenen Anwendungsfällen mindestens in ähnlicher Form existieren, ausgewählt, kombiniert und als abstrakter Anwendungsfall beschrieben. Lenz und Kuhn stellen hierzu fest, dass bei der IT-Unterstützung abteilungsübergreifender Abläufe wiederkehrende Grundmuster, wie z. B. die Auftragserteilung, aber auch die Befund-Rückübermittlung vorkommen (Lenz & Kuhn 2004). Diese Grundmuster sind ebenfalls in den Anwendungsfällen aus Kapitel 2.3.1 zu finden. Bei genauer Betrachtung sind noch umfangreichere Gemeinsamkeiten erkennbar. In allen beschriebenen Prozessen wird der Vorgang durch den Eingang eines internen oder externen klinischen Auftrags initiiert, woraufhin der zuständige Leistungserbringer die Anamnese (Krankengeschichte) des Patienten erhebt. Im Anschluss daran werden eine oder mehrere Leistungen erbracht. Diese können diagnostische oder behandelnde Leistungen sein. Nach oder bereits während der Erbringung der Leistungen werden diese dokumentiert und in einem zusammenfassenden Dokument (z. B. Arztbrief oder Befunddokument) festgehalten. Erfordert mindestens eine der erbrachten Leistungen eine Anschlussleistung einer anderen Leistungsstelle, so wird diese angefordert. Abschließend wird der Patient aus dem aktuellen Teilprozess entlassen und die entstandene Dokumentation dem Auftraggeber und/oder einem nachfolgenden Leistungserbringer zur Verfügung gestellt.

Zudem ist zu berücksichtigen, dass auf dieser Betrachtungsebene medizinische Leistungen nicht zwangsläufig direkt an einem Patienten ausgeführt werden. So kann der Untersuchungsgegenstand beispielsweise bei einer Laboruntersuchung eine Blut- oder Urinprobe, bei einer pathologischen Untersuchung eine Gewebeprobe usw. sein. Bei einer teleradiologischen Untersuchung kann die erbrachte Leistung die Untersuchung einer elektronisch übermittelten Röntgenaufnahme sein.

Bei Teleradiologie handelt es sich lt. § 2 Abs. 24 RöV um die *„Untersuchung eines Menschen mit Röntgenstrahlung unter der Verantwortung eines Arztes nach § 24 Abs. 1 Nr. 1, der sich nicht am Ort der technischen Durchführung befindet und der mit Hilfe elektronischer Datenübertragung und Telekommunikation insbesondere zur rechtfertigenden Indikation und Befundung unmittelbar mit den Personen am Ort der technischen Durchführung in Verbindung steht.“*

Trotz dieser einschränkenden Feststellung wird für die Beschreibung der Abläufe des abstrakten Anwendungsfall der Akteur Patient verwendet. Er schließt folglich Proben und Bilder vom Patienten ein.

Die Darstellung des abstrakten Anwendungsfalls ist so gewählt, dass es unerheblich ist, ob ein Auftrag von einem internen oder externen Auftraggeber gestellt wurde, und dass prinzipiell alle möglichen Auftrags- und Leistungskombinationen durch Konkretisierung abgebildet werden können.

Zur Verbesserung der Prägnanz wird der abstrakte Anwendungsfall aus zwei verschiedenen Perspektiven betrachtet. Die erste Betrachtungsperspektive ist die der inneren Betrachtung aus der Sicht einer einzelnen Leistungsstelle. Dabei wird davon ausgegangen, dass sich der Teilprozess, der innerhalb einer Leistungsstelle abläuft, so darstellen lässt, dass er für jede im Gesamtprozess enthaltene Leistungsstelle gültig ist. In der zweiten Perspektive wird eine Betrachtung von außen vorgenommen. Das bedeutet, dass der Fokus auf der Verbindung zwischen den einzelnen Leistungsstellen liegt.

Anwendungsfall: Betrachtung aus Sicht einer Leistungsstelle (interne Sicht)

Beteiligte Akteure:

- Alle aus Kapitel 2.3.1.1.
- Bei Auftraggebern wird zwischen extern und intern unterschieden
- Bei Leistungserbringern gibt es neben den internen Leistungserbringern auch noch einen nachfolgenden Leistungserbringer.
- Beim Akteur Patient kann es sich um den Patienten selbst, aber auch andere Untersuchungsobjekte wie Proben oder Aufnahmen vom Patienten handeln.

Auslöser: Eine Anforderung einer externen Stelle und der zugehörige Patient treffen bei der aktuellen Leistungsstelle ein.

Vorbedingungen: Alle Bedingungen, die für die Durchführung von Leistungen an der aktuellen Leistungsstelle notwendig sind, werden erfüllt. Das betrifft sowohl den Zustand des Patienten bzw. der Probe als auch den Inhalt der externen Leistungsanforderung.

Beschreibung: Nach dem Eintreffen des Patienten oder des Untersuchungsgegenstands bei der Leistungsstelle wird durch den dort verantwortlichen Leistungserbringer mit der nachfolgenden Leistung angemessenen Verfahren die Anamnese des Patienten erhoben.

Aufbauend auf diesen Erkenntnissen werden anschließend die nötigen Parameter für die zu erbringende Leistung bestimmt und die in der Leistung enthaltenen Maßnahmen durchgeführt. Nach der Leistungserbringung wird geprüft, ob aufgrund von deren Ergebnis die interne oder externe Durchführung einer weiteren Leistung erforderlich ist. Ist das der Fall, so werden dafür eine Anforderung gestellt und die Gründe für ihre Anordnung dokumentiert. Handelt es sich um eine Leistung, die von einer externen Leistungsstelle erbracht werden muss, so wird die Anforderung an die betreffende Leistungsstelle übermittelt und die zuvor erbrachte Leistung dokumentiert. Sind alle Leistungen, die innerhalb der betreffenden Leistungsstelle erbracht werden mussten, abgeschlossen, wird ein zusammenfassendes Befunddokument erstellt. Bei der Umsetzung in konkreten Anwendungsfällen kann die Reihenfolge der Anforderung der nachfolgenden Leistung und der Dokumentation variieren.

Nachbedingungen: Alle angeforderten Leistungen sind abgeschlossen oder aufgrund eines während der Erhebung der Anamnese festgestellten Befunds als nicht sinnvoll identifiziert. Die entstandenen Ergebnisse wurden dokumentiert und in einer systematisch angelegten Akte für deren weitere Verwendung archiviert.

Exemplarischer Ablauf in Aktivitätsdiagrammform:

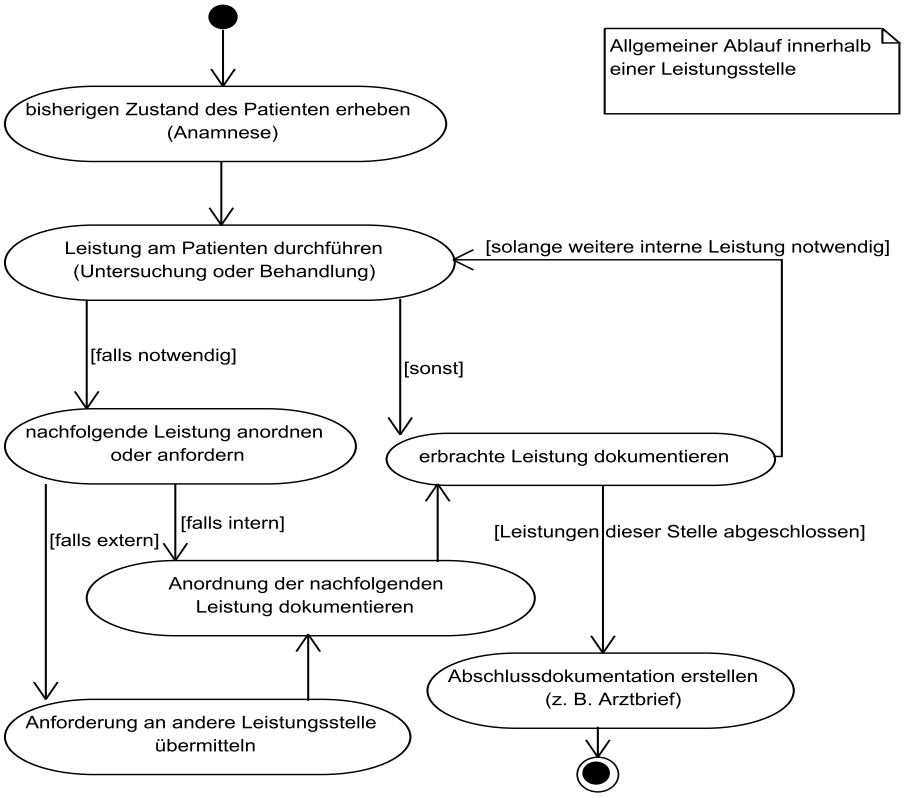



Abbildung 19: Aktivitätsdiagramm: Allgemeiner Ablauf des Prozesses innerhalb einer Leistungsstelle

Anwendungsfall: Betrachtung des Gesamtprozesses (externe Sicht)

Beteiligte Akteure:

- Alle aus Kapitel 2.3.1.1.
- Beim Akteur Patient kann es sich um den Patienten selbst, aber auch andere Untersuchungsobjekte wie Proben oder Aufnahmen vom Patienten handeln.

Auslöser: Ein Patient mit einer Erkrankung betritt durch sein Eintreffen in einer Versorgungseinrichtung einen leistungsstellen- oder einrichtungsübergreifenden Behandlungspfad.

Vorbedingungen: Für die Erkrankung des Patienten ist in den betroffenen Einrichtungen ein leistungsstellen- oder einrichtungsübergreifender Behandlungspfad definiert.

Beschreibung: Ausgehend von der Annahme, dass sich die Arbeitsschritte vor und nach der Erbringung einer Leistung – sofern sie in ausreichend abstrakter Form dargestellt sind – immer gleichen, wird im Aktivitätsdiagramm zu diesem Anwendungsfall der Behandlungspfad als rekursiver Prozess abgebildet. Er beginnt mit dem Eintreffen des Patienten bei der ersten Leistungsstelle des Behandlungspfades. Danach erfolgt die Erhebung von Daten zum Zustand des Patienten. Dabei kann es sich sowohl um die Ermittlung von Daten aus früheren Leistungen als auch um eine vorgenommene Eingangsuntersuchung handeln. Dieser Schritt kann übersprungen werden, sollte es sich bei dem Akteur Patient um

eine Probe- oder Aufnahme zur Befundung handeln. Anschließend wird, abhängig von den ermittelten Parametern, die eigentliche Leistung am Patienten erbracht und deren Ergebnis dokumentiert. Sämtliche innerhalb der gleichen Leistungsstelle abzuwickelnden Folgeleistungen werden entsprechend der „Betrachtung aus Sicht einer Leistungsstelle“ behandelt und hier nicht weiter erläutert. Handelt es sich um die letzte Leistung, die im zugehörigen Behandlungspfad vorgesehen ist, so wird der Patient anschließend aus dem Behandlungsprozess entlassen. Schreibt der Behandlungspfad eine Folgeleistung vor, so wird diese bei der betreffenden externen Leistungsstelle angefordert. Die Anforderung einer Leistung setzt eine Datenübertragung voraus. Die Übertragung der Leistungsanforderung kann allerdings auch über den Patienten, der sich seinen nächsten Leistungserbringer frei auswählt, erfolgen. Anschließend wird die entstandene Dokumentation der nachfolgenden Leistungsstelle zugänglich gemacht. Das kann sowohl durch aktives Zusenden der Dokumente als auch durch eine Bereitstellung zur Abholung erreicht werden. Nach der Übermittlung begibt sich der Patient zur nächsten Leistungsstelle, oder er wird dorthin transportiert. Dadurch beginnt der beschriebene Vorgang von Neuem.

Nachbedingungen: Alle medizinisch erforderlichen Schritte des Behandlungsprozesses sind durchlaufen. Das schließt auch eine vorzeitige Beendigung des Behandlungspfades durch eine eigenständige Entlassungsentscheidung des Patienten oder durch dessen Tod ein. Alle im Rahmen des Prozesses durchgeführten Leistungen, deren Ergebnisse sowie der Zustand des Patienten zu verschiedenen Zeitpunkten und sein Weg durch eine Vielzahl von Leistungsstellen sind elektronisch dokumentiert.

Exemplarischer Ablauf in Aktivitätsdiagrammform:

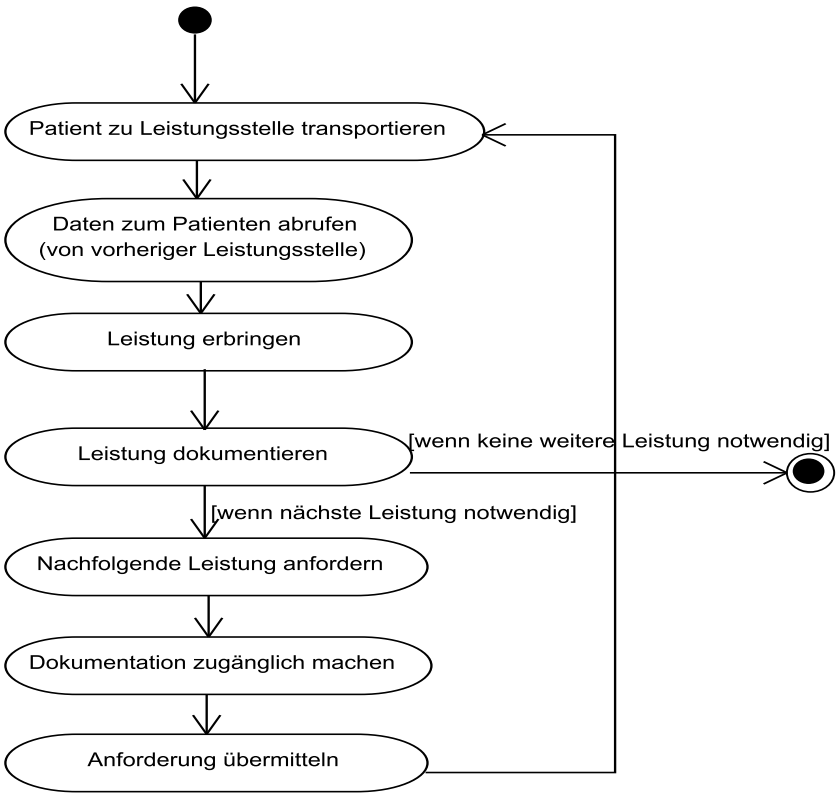



Abbildung 20: Aktivitätsdiagramm: Abstrakter Anwendungsfall - externe Sicht

Die zweite der beiden abstrakten Anwendungsfallbeschreibungen zeigt die elektronische Abfrage der bereits bestehenden medizinischen Dokumentation, die elektronische Übermittlung von Leistungsanforderungen und die Erteilung von Zugriffsrechten auf die neu entstandene medizinische Dokumentation oder deren Übermittlung an nachfolgende Leistungserbringer. Dabei handelt es sich um Schritte, zu deren Umsetzung zwangsläufig auf Mittel der entfernten Kommunikation in verteilten Systemen zurückgegriffen werden muss. Diese Notwendigkeit gründet auf der Tatsache, dass auf entfernte Kommunikation nur in einem monolithischen System verzichtet werden kann. Ein monolithisches System, das alle medizinischen Versorgungseinrichtungen mit allen ihren Leistungsstellen beinhaltet, ist aufgrund der großen Funktionsvielfalt aus heutiger Sicht sowohl technisch als auch organisatorisch und politisch nicht realisierbar.

Mayr bezeichnet die Übergänge zwischen verschiedenen Versorgungseinrichtungen als potenzielle Bruchstellen (Mayr 2010) und verdeutlicht die daraus resultierenden Probleme anhand der häufigen Wechsel älterer Patienten zwischen Seniorenheimen und Krankenhäusern. Der beschriebene Ablauf entspricht dabei einer Leistungsanforderung, die durch einen Leistungserbringer in einem Seniorenheim gestellt wird und an eine Leistungsstelle in einem Krankenhaus adressiert ist. Er belegt damit, dass gerade der Bereich der Schnittstellen ein noch immer nicht flächendeckend zufriedenstellend gelöstes Problem bei der effizienten Unterstützung einrichtungsübergreifender Behandlungspfade ist.

Beide dargestellten abstrakten Behandlungspfade sind zu dem Referenzmodell HL7 RIM kompatibel, das ein abstraktes Klassenmodell zur Dokumentation von medizinischen Leistungen definiert (vgl. Kapitel 2.5.4).

## 2.4. Anforderungen der Systembetroffenen

Die Entwicklung von Software ist kein rein technischer Prozess. Verschiedene soziale Aspekte innerhalb der Zielgruppe des Softwaresystems beeinflussen ihn ganz wesentlich (Oestereich 2009:S. 16). Aber auch ohne Berücksichtigung der Auswirkungen sozialer Aspekte beeinflusst die Vorstellung der Benutzer oder allgemein der Systembetroffenen, vom gewünschten Zielzustand des Systems entscheidend dessen spätere Erscheinung und seinen internen Aufbau.

Bei der Betrachtung der Anforderungen wird üblicherweise zwischen funktionalen und nichtfunktionalen Anforderungen unterschieden (Partsch 2010:S. 27; Sommerville 2007:S. 152 ff.).

### 2.4.1. Funktionale Anforderungen

Bei den funktionalen Anforderungen handelt es sich um Anforderungen, die die nutzbaren Funktionen eines Produktes beschreiben. Somit beschreiben sie, welche Aufgaben mit dem System erfüllt werden müssen (Partsch 2010:S. 27). Eine funktionale Anforderung für eine elektronische Krankenakte kann beispielsweise wie folgt lauten: „Mit dem System müssen Arztbriefe erfasst werden können“. In Kapitel 2.3 sind exemplarisch und allgemeingültig die Abläufe im System einer verteilten Krankenakte in Form von je einer ausgefüllten Anwendungsfallschablone und einem Aktivitätsdiagramm beschrieben. Dabei beinhaltet jeder Prozessschritt eine einzelne funktionale Anforderung an das System. Folglich lässt sich bei näherer Betrachtung erkennen, dass die in Kapitel 2.3 betrachteten Anwendungsfälle bereits die funktionalen Anforderungen an verteilte Krankenakten behandeln.

In Anlehnung an Schramm-Wölk und Schug (Jähn & Nagel 2004:S. 18) lassen sich die funktionalen Anforderungen aus technischer Sicht in die folgenden allgemeinen Gruppen zusammenfassen:

- Erstellen und Bearbeiten von elektronischer Dokumentation
- Speichern und Archivieren der elektronischen Dokumentation
- Suchen, Durchsuchen und Anzeigen elektronischer Dokumentation
- Auswerten der enthaltenen Daten
- Elektronisches Übermitteln von Inhalten.

Die Art und Struktur der Daten sowie die exakte Beschreibung der Anforderung variiert dabei abhängig vom tatsächlichen Anwendungskontext im Rahmen des konkreten Versorgungsprozesses. Allgemeiner betrachtet lassen sich die funktionalen Anforderungen auf das suchen, finden, erfassen, speichern, zugängig machen und übertragen von Ergebnisdokumenten und Aufträgen zusammenfassen.

### 2.4.2. Nichtfunktionale Anforderungen

Zusätzlich zu den funktionalen müssen bei der Konzeption und Entwicklung von IT-Systemen auch nichtfunktionale Anforderungen berücksichtigt werden (Partsch 2010:S. 27–28). Sie beschreiben qualitative oder sonstige zusätzliche Eigenschaften des Systems.

Beispiele für nichtfunktionale Anforderungen sind die Systemverfügbarkeit, die Sicherheit des Systems, dessen Zuverlässigkeit, seine Erweiterbarkeit oder die nachträgliche Nachvollziehbarkeit durchgeführter Systemhandlungen. Auch Anforderungen bezüglich der Benutzerfreundlichkeit oder generell an die Qualität der Schnittstellen für die Mensch-Maschine-Interkation sind solche nichtfunktionalen Anforderungen. Nichtfunktionale Anforderungen beeinflussen wesentlich die technische Architektur eines Systems und die Auswahl der verwendeten Design-Konzepte.

Bezogen auf verteilte Krankenakten haben sich verschiedene Autoren mit der Identifikation von nichtfunktionalen Anforderungen für diesen spezifischen Bereich sowie mit der Begründung ihrer Relevanz beschäftigt. Ihre Erkenntnisse sollen im folgenden Abschnitt zu einer zusammengehörigen Sammlung vereint werden, um in den nachfolgenden Kapiteln beim Aufbau der Struktur bzw. Darstellungsform einer allgemeinen Entwicklungsmethodik für verteilte Krankenakten herangezogen werden zu können.

Brunner und Kaiser beschreiben die folgenden, durch die Konferenz der Datenschutzbeauftragten des Bundes und der Länder geprägten nichtfunktionalen Anforderungen für elektronische Krankenakten:

„[…]:

- *Vertraulichkeit (Nur Befugte dürfen patientenbezogene Daten zur Kenntnis nehmen)*
- *Authentizität (Der Urheber bzw. der Verantwortliche muss eindeutig feststellbar sein)*
- *Integrität (Patientendaten müssen unversehrt, vollständig, gültig und widerspruchsfrei sein)*
- *Verfügbarkeit (Daten müssen zeitgerecht zur Verfügung stehen)*
- *Revisionsfähigkeit (Die Verarbeitungsprozesse müssen nachvollziehbar sein)*
- *Validität (Die Verarbeitungsqualität muss für den Nutzungszweck angemessen sein)*
- *Rechtssicherheit (Nachweise der Verarbeitung müssen Beweiskraft haben)*
- *Nicht-Abstreitbarkeit von Datenübermittlungen*
- *Nutzungsfestlegung (Ein Berechtigungskonzept muss vorhanden sein)“* (Brunner & Kaiser 2008:S. 72).

Schramm-Wölk und Schug (Schramm-Wölk & S. H. Schug 2004:S. 18–20) identifizieren die Bereiche Rechtswesen, Datenschutz, Sicherheit und Kommunikation als wesentliche Treiber für nichtfunktionale Anforderungen. Darauf aufbauend identifizieren sie die Verfügbarkeit, die Transparenz der Verfahrensweise bei der Verarbeitung der Daten, die Integrität der Daten sowie deren Vertraulichkeit und die Authentizität der Benutzer als zentrale nichtfunktionale Anforderungen. Des Weiteren fordern sie eine hohe Integrationsfähigkeit in existierende Arbeitsplatzsysteme sowie Schnittstellenstärke. Bei Schnittstellenstärke handelt es sich um die Fähigkeit, eine Vielzahl von Schnittstellen-Standards zu unterstützen. Dadurch erweitern Schramm-Wölk und Schug die Auflistung von Brunner und Kaiser um die Anforderung, den Aspekt der Verteilung flexibel und für die Benutzer transparent zu gestalten.

Der Begriff *Gesundheitstelematik* ist ein gängiges Kunstwort aus den Begriffen *Gesundheitswesen*, *Telekommunikation* und *Informatik*. Es bezeichnet allgemein einrichtungsübergreifend verteilte Systeme in der Medizin (Haas 2006:S. 3–4). Peter Haas listet in seinem Buch

„Gesundheitstelematik“, in den Kapiteln 2.2.2 bis 2.2.5 über die Grundlagen zur Gesundheitstelematik (Haas 2006:S. 40–55), eine Reihe notwendiger Eigenschaften für verteilte Systeme mit einem Anwendungsgebiet im Gesundheitswesen auf.

Er gliedert diese Eigenschaften in die Bereiche:

- Kommunikationsmerkmale,
- technische Aspekte und
- datenschutzrechtliche Aspekte.

Während Haas bei den Kommunikationsmerkmalen deren mögliche Ausprägungsformen benennt und diese als Möglichkeit zur Beschreibung der gewünschten Kommunikation in einem konkreten System anbietet, handelt es sich bei seinen technischen und datenschutzrechtlichen Aspekten um konkrete Anforderungen, also geforderte Ausprägungsformen, die an ein Telematik-System und folglich auch an eine verteilte Krankenakte zu stellen sind. Entsprechend besteht bei den Kommunikationsmerkmalen für eine konkrete verteilte Krankenakte eine Wahlfreiheit bezüglich der gewünschten Merkmale der enthaltenen Kommunikationsbeziehungen. Diese müssen allerdings im Zuge der Planung eines IT-Systems für eine verteilte Krankenakte ausgewählt und als konkrete Anforderungen definiert werden, um das Ziel-System umsetzbar zu beschreiben.

Bei den technischen Aspekten führt Haas die Qualität des Übertragungsmediums und dessen Datendurchsatz als wesentliche Größen an. Die Qualität des Übertragungsmediums ist die Eigenschaft der unverfälschten Übertragung von Nachrichten. Der Datendurchsatz muss abhängig von der konkreten Anwendung in der verteilten Krankenakte als Anforderung quantifiziert werden. Die Anforderungen an den Datendurchsatz sind hierbei von der Art der übertragenen Daten abhängig. Eine hochauflösende CT- oder Röntgenaufnahme benötigt einen höheren Datendurchsatz als ein Dokument mit textuellem Befund. Weitere technische Aspekte sind die eindeutige Adressierbarkeit der Systemteilnehmer mittels eines geeigneten Teilnehmerverzeichnisses und effiziente Mechanismen für den Verbindungsaufbau und das Routing, um eine zuverlässige und gleichermaßen zügige Übertragung der Daten zwischen den Teilnehmern zu gewährleisten.

Als datenschutzrechtliche Aspekte führt Haas (Haas 2006:S. 50–55) die

- Vertraulichkeit der Kommunikation,
- Authentizität des Senders und Empfängers,
- Authentizität der Nachricht,
- Integrität der Kommunikationsinhalte,
- Unabstreitbarkeit der Richtigkeit und der Übermittlung sowie
- Justiziabilität, also Überprüfbarkeit durch die Rechtsgültigkeit der Signatur an.

Hörbst und andere (Hoerbst u. a. 2009:S. 387, Tabelle 2) ermittelten durch eine Expertenbefragung Anforderungen, die eine elektronische Patientenakte im Rahmen einer möglichen zukünftigen Zertifizierung erfüllen muss. Die befragten Experten identifizierten die folgenden Punkte als für die Qualität einer elektronischen Patientenakte besonders wichtig:

- Datenschutz
- Datensicherheit
- Benutzerfreundlichkeit (Usability)
- Zugängigkeit (Accessibility)
- Verfügbarkeit
- Funktionsumfang (d. h. Erfüllung der funktionalen Anforderungen)
- Fehlertoleranz
- Unterstützung der real existierenden Prozesse (funktionale Anforderung)
- Skalierbarkeit
- Stabilität
- Wartungsfreundlichkeit und
- Veränderbarkeit bzw. Anpassbarkeit.

Dabei sind die Anforderungen zum Funktionsumfang und zur Unterstützung der Prozesse und Workflows in dieser Auflistung die Repräsentanten der funktionalen Anforderungen, während die anderen Punkte Gruppen nichtfunktionaler Anforderungen darstellen. Zusammenfassend betrachtet, sind in den Aussagen der in diesem Kapitel zitierten Autoren die folgenden nichtfunktionalen Anforderungen enthalten:

***Sicherheit und Datenschutz***

- Vertraulichkeit
- Authentizität
- Integrität
- Nichtabstreitbarkeit
- Revisionsfähigkeit und Rechtssicherheit
- Nutzungsfestlegung und Berechtigungskonzept

***Zuverlässigkeit***

- Verfügbarkeit
- Validität
- Revisionsfähigkeit
- Verbindungsqualität
- Fehlertoleranz
- Skalierbarkeit
- Stabilität

***Erweiterbarkeit, Veränderbarkeit und Anpassbarkeit***

- Skalierbarkeit
- Schnittstellenstärke
- Integrierbarkeit

***Kommunikation***

- Vertraulichkeit
- Zuverlässigkeit
- Übertragungskapazität
- Kommunikationsform (synchron vs. asynchron).

Die ermittelten Anforderungen werden in Kapitel 4 in die Auswahl der Schwerpunkte einer geeigneten Pattern-Sprache für die Entwicklung verteilter Krankenakten einfließen.

Auch wenn die Benutzerfreundlichkeit in der Medieninformatik eine zentrale Rolle spielt und in der Auflistung von Hörbst erwähnt wird, findet sie hier nicht Eingang in die Auflistung der Schwerpunkte der ermittelten nichtfunktionalen Anforderungen. Das ist damit zu begründen, dass Benutzerinteraktion kein Aspekt ist, der für verteilte Krankenakten spezifisch ist. Der Aspekt der Verteilung ist unabhängig vom Aspekt der Benutzerinteraktion. Annahmen, Anforderungen und Lösungen, die die Interaktion zwischen Benutzern und verteilten Krankenakten betreffen, sind in exakt gleicher Weise auch für beliebige nicht verteilt realisierte elektronische Krankenakten gültig. Das gilt umso mehr, zumal für verteilte Krankenakten (siehe S. 38) eine möglichst weitreichende Verteilungstransparenz gefordert wird.

## 2.5. Standards

Standards regeln die Einheitlichkeit technischer Produkte um deren Gleichheit bezüglich einer oder mehrerer definierter Eigenschaften zu erreichen. Ihr Zweck ist es z. B., die gegenseitige Kompatibilität von Produkten unabhängig von einzelnen Herstellern zu gewährleisten. Sie sind somit Mittel zur Vereinheitlichung von Produkten oder bestimmten Eigenschaften von Produkten. Die Standardisierung eines Sachverhalts ist an komplexe Verfahren gebunden und nimmt üblicherweise mehrere Jahre in Anspruch (Clement & Schreiber 2010:S. 209). Dabei wird zwischen regulativen Standards und koordinativen Standards unterschieden (siehe z. B. Clement & Schreiber 2010:S. 210, Abbildung 8.1). Regulative Standards sind zumeist staatlich auferlegte Gebote bzw. Verbote. Sie können z. B. zur Erreichung von bestimmten qualitativen Mindestanforderungen bei Produkten einer Gattung eingesetzt werden. Koordinative Standards liefern verfügbare Regeln, die von Herstellern genutzt werden können, um ihre Produkte zu anderen Produkten kompatibel zu gestalten.

Kompatibilität zu verschiedenen Systemen bereits existierender Infrastruktur ist ein wesentliches Kriterium für die Einsetzbarkeit von zusätzlichen IT-Systemen in Unternehmen, Behörden oder Kliniken. Kompatibilität beinhaltet in diesem Fall vor allem die Fähigkeit, dass unterschiedliche Systemkomponenten miteinander erfolgreich kommunizieren können. Diese Form der Kompatibilität wird häufig auch Interoperabilität genannt. Kommunikation ist dabei der Austausch von Information. Erfolgreiche Kommunikation wiederum besteht aus drei wesentlichen Bestandteilen. Grundlegend ist hierbei das Vorhandensein eines gemeinsamen Kommunikationskanals. Außerdem werden eine beiden Kommunikanten gemeinsame Grammatik oder Datenstruktur sowie eine eindeutige Definition des verwendeten Vokabulars, also der Werte und ihren Einheiten innerhalb der Datenstruktur benötigt.

Schug und Schramm-Wölk (S. H. Schug & Schramm-Wölk 2004:S. 11) klassifizieren Standards für die Interoperabilität von elektronischen Patientenakten deshalb anhand der Ziele, die durch sie erreicht werden sollen. Diese Ziele sind:

- technische Interoperabilität,
- syntaktische Interoperabilität und
- semantische Interoperabilität.

Technische Interoperabilität wird vorwiegend durch allgemeine IT-Standards erreicht. Sie sind nicht spezifisch für elektronische Krankenakten und umfassen alle technischen Voraussetzungen, die zur zuverlässigen Übertragung von Daten zwischen zwei Systemen benötigt werden. Ihr Spektrum reicht von zueinander kompatibler Netzwerkhardware über gemeinsame Protokolle bis hin zu gemeinsamen Zeichensätzen. Technische Interoperabilität setzt auch eine Kompatibilität der Sicherheitsmaßnahmen der Übertragungsebene voraus. Das bedeutet, dass beispielsweise die verwendeten Authentifizierungs- und Verschlüsselungsmechanismen zueinander kompatibel sein müssen. Technische Interoperabilität umfasst folglich die Fähigkeit von Systemen zielgerichtet Daten austauschen zu können.

Ist bei zwei Systemen die technische Interoperabilität erreicht, ist der nächste logische Schritt die Herstellung syntaktischer Interoperabilität. Diese wird zwischen IT-Systemen durch die Verwendung eines gemeinsamen Formats zur Datenübermittlung erreicht. Dabei definiert das Format, ähnlich einer Grammatik, die Bedeutung eines Feldes in Abhängigkeit von dessen Position innerhalb der Menge der übermittelten Daten. So wird beispielsweise durch das Format festgelegt, wo sich der Name eines Patienten innerhalb der übermittelten Daten befindet oder wie dessen Position ermittelt werden kann.

Technische- und syntaktische Interoperabilität sind eine notwendige Voraussetzung für die erfolgreiche Kommunikation zwischen zwei Systemen. Sie sind allerdings nicht ausreichend, um eine fehlerfreie automatisierte Weiterverarbeitung übermittelter Daten zu gewährleisten. Dazu ist es erforderlich, dass die Systeme auch auf semantischer Ebene kompatibel sind. Das heißt, dass die Bedeutung der einzelnen möglichen Werte innerhalb der Felder des verwendeten Formats eindeutig definiert ist. Bei numerischen Messwerten geschieht das durch die Definition der Einheit der enthaltenen Zahlenwerte. Bei anderen Feldtypen kann das zum Beispiel durch die Festlegung eines klar definierten Vokabulars in Form von standardisierten Aufzählungstypen erfolgen.

Anders als Schug und Schramm-Wölk unterteilt Haas (Haas 2006:S. 295 ff.) die für Gesundheitstelematik und folglich für verteilte Krankenakten relevanten Standards in die Gruppen:

- Kommunikationsstandards,
- Dokumentenstandards,
- Datenmodellstandards oder Referenzmodelle,
- Architekturstandards und
- Semantikstandards.

Er betrachtet die relevanten Standards im Gegensatz zu Schug und Schramm-Wölk nicht ausschließlich bezogen auf den Aspekt des Informationsaustauschs. Durch die Gruppen *Datenmodellstandards* und *Architekturstandards* sieht er auch Standards vor, die sich mit Interna der am Informationsaustausch beteiligten Systemknoten befassen. Diese Betrachtungsweise geht daher über die Standardisierung von Schnittstellen zur Kommunikation hinaus.

Mit den Gruppen *Kommunikationsstandards*, *Dokumentenstandards* und *Semantikstandards* behandelt Haas die technische, syntaktische und semantische Interoperabilität. Dabei dient die Gruppe der Kommunikationsstandards der Behandlung von Mechanismen zur Erreichung technischer Interoperabilität. Sie befasst sich mit der grundsätzlichen Fähigkeit, Daten zwischen zwei oder mehreren technischen Systemen auszutauschen. Kommunikation beinhaltet den Austausch von Information zwischen zwei oder mehreren Kommunikanten (siehe 2.1.8). Folglich werden alle Standards, die sich mit der Übertragung von Daten oder dem Austausch von Nachrichten zwischen Systemen befassen, als Kommunikationsstandards bezeichnet.

Die Dokumentenstandards befassen sich mit der Struktur von übertragbaren Objekten, den Dokumenten. Durch die Festlegung der Bedeutung der in einem Objekt enthaltenen Attribute oder Felder definieren sie eine eindeutige Syntax für Dokumente, die als Nachrichten, Parameter oder Rückgabewerte im Rahmen der Kommunikation zwischen medizinischen Informationssystemen übertragen werden können. Die Gruppe der Semantikstandards ist sowohl inhaltlich als auch namentlich mit den Standards für die semantische Interoperabilität von Schug und Schramm-Wölk identisch.

Datenmodellstandards oder Referenzmodelle betreffen ebenso wie die Dokumentenstandards Strukturen zur Ablage und Verwaltung von Daten. Sie fokussieren allerdings nicht das Dokument als einfache Entität, sondern die Abbildung aller für die Zieldomäne relevanten Daten und Strukturen in Form eines umfassenden Datenmodells. Das Ziel dieser Standards liegt folglich auf der Standardisierung von internen Datenstrukturen für eine Gattung von Informationssystemen (wie z. B. Krankenhaus-Informationssystemen).

Die Architekturstandards bilden eine Gruppe, deren Name leicht zu Missverständnissen bei der Interpretation der damit verbundenen Zielsetzung führen kann. Der Begriff *Architektur* findet in der Informatik in unterschiedlichen Ausprägungsformen Verwendung (siehe S. 28). In dieser Arbeit werden Standards, die ein zusammenhängendes Konzept aus der Richtung mehrerer Schwerpunkte betrachten und so eine Art Gesamtarchitektur für die Krankenakte beschreiben, als Architekturstandards bezeichnet.

Bei Architekturstandards ist der Nutzen einer verbindlichen Standardisierung bisher nicht eindeutig geklärt. Denn Standardisierung kann durch die Einschränkung des Gestaltungsspielraums bezüglich der Softwarearchitektur (Haas 2006:S. 296) positive Einflüsse durch neue Methoden und Erkenntnisse aus dem Software-Engineering verhindern. Andererseits kann damit aber die Erreichung einer gewissen Mindestqualität sichergestellt werden. Dazu ist aber bereits zum Zeitpunkt der Standardisierung umfassendes Wissen über die bei der Entwicklung und dem Betrieb eines entsprechenden Systems auftretenden Einflussfaktoren nötig. Dieses Wissen ist bei verteilten Krankenakten noch nicht in ausreichendem Umfang gegeben (Glesner u. a. 2008:S. 128), weshalb Architekturstandards für die weitere Betrachtung nur als zusätzliche Referenzmodelle, aber nicht als im Sinne normativer Standards relevante Gruppe betrachtet werden. Haas (Haas 2006:S. 362 ff.) beschreibt Architekturstandards gemeinsam mit Referenzmodellen, denn die von ihm angeführten Architekturstandards beinhalten auch generische Ansätze zur Umsetzung geeigneter Datenstrukturen und werden dadurch bei ihrer praktischen Anwendung in einem der Verwendung von Referenzmodellen ähnlichen Kontext genutzt.

Unter Berücksichtigung der beiden genannten Klassifizierungsschemata für Standards die elektronische Patientenakten bzw. die Gesundheitstelematik beeinflussen, sowie den genannten Einschränkungen, werden die für verteilte Krankenakten relevanten Standards den Gruppen *Kommunikationsstandards*, *Dokumentenstandards*, *Semantikstandards* sowie *Referenzmodelle und Architekturstandards* zugeordnet.

### 2.5.1. Kommunikationsstandards

Als Kommunikationsstandards werden zusammenfassend die Standards betrachtet, die den Austausch von Nachrichten zwischen IT-Systemen ermöglichen. Sie werden nachfolgend in zwei Gruppen unterteilt. Eine Gruppe umfasst jene Standards, die technisch die Übertragung von Daten zwischen Computern ermöglichen. Sie werden Übertragungsstandards genannt. Die andere Gruppe befasst sich mit Kommunikationsstandards zur Übertragung strukturierter Nachrichten, Request-Reply-Protokollen und entfernten Funktionsaufrufen.

Übertragungsstandards sind Standards, die eine Übertragung von Daten von einer Entität zu einer oder mehreren anderen vereinheitlichen und dadurch möglich machen. Die interne Struktur und der Informationsgehalt der übermittelten Daten sind dabei nicht relevant. Übertragungsstandards dienen als technische Grundlage für die Kommunikation zwischen zwei oder mehreren Systemknoten. Hierbei ist die Adressierbarkeit von Systemknoten eine notwendige Voraussetzung für die gezielte Auswahl von Kommunikationspartnern.

Zur Übertragung von Daten dienen allgemeine IT-Standards, wie z. B. die Standards aus dem Internet-Protokoll-Stack (Kurose & Ross 2008:S. 73). Sie sind nicht spezifisch für den Einsatz in medizinischen Informationssystemen konzipiert, werden allerdings als Grundlage für die Kommunikation zwischen den Bestandteilen verteilter Krankenakten genutzt.

Derzeit typische Vertreter von Standards für die Ebenen des Internet-Protokoll-Stacks sind:

- Physical Layer: Ethernet[4], ATM, UMTS usw.,
- Network und Data Link Layer: Internet Protocol (IP)[5] und
- Transport Layer: Transmission Control Protocol (TCP)[6].

Miteinander kombiniert eröffnen sie die Möglichkeit zu synchronisiertem Datenaustausch über Datenströme. Vor allem die Protokolle der physikalischen Schicht (Physical Layer) sind durch die unterschiedlichen Übertragungswege mittels Kabel und Funk sowie über kurze und lange Strecken sehr vielfältig.

Über die rein technische Übertragung hinaus bieten Request-Reply-Protokolle, Messaging-Protokolle und Protokolle für entfernte Methoden- oder Funktionsaufrufe als Protokolle der Anwendungsschicht Dienste zur Übertragung von Informationen an. Sie verwenden den oben genannten Protokoll-Stack und übertragen damit Nachrichten oder Parameter aus strukturierten Daten. Eine Standardisierung der Inhalte der strukturierten Nachrichten und Parameter kann durch die Anwendung von Dokumentenstandards erreicht werden.

Beispiele für Protokolle der Anwendungsschicht sind z. B. der E-Mail-Dienst mit SMTP und POP3 bzw. IMAP zum asynchronen Austausch von Nachrichten oder das Hypertext-Transfer-Protokoll HTTP für die synchrone Request-Reply-Kommunikation zwischen Webservern und Browsern. Die genannten Protokolle erlauben die Übertragung von Daten zwischen zwei oder

[4] Standarddokumente siehe: http://standards.ieee.org/about/get/802/802.3.html
[5] Ursprüngliches Standarddokument siehe: http://tools.ietf.org/html/rfc791
[6] Ursprüngliches Standarddokument siehe: http://tools.ietf.org/html/rfc793

mehreren Kommunikanten, die durch symbolische Namen identifiziert werden, und legen der Übertragung ein definiertes Kommunikationsprotokoll zugrunde.

Verschiedene Kommunikationsmechanismen zwischen medizinischen IT-Systemen bauen auf den dargestellten Protokollen und Übertragungsstandards auf. Der Teil 8 des DICOM Standards (Network Communication) (NEMA 2009c) ist ein Standard für die Übertragung von Dateien bildgebender Modalitäten, der direkt auf TCP/IP aufbaut. Er ist als Übertragungsstandard speziell für die Anwendungsfälle der medizinischen Bildgebung und Bildübertragung konzipiert und beinhaltet standardisierte Operationen wie C-FIND oder C-MOVE, mit denen DICOM-Serien in Bildarchiven gefunden bzw. zwischen DICOM-fähigen Systemen übertragen werden können. DICOM-Serien sind zusammengehörige Bildsammlungen einschließlich der zugehörigen Metadaten. Sie könnten auch als DICOM-Dateien bezeichnet werden. Die an der Kommunikation beteiligten Endgeräte werden DICOM-ApplicationEntitys (AEs) genannt. Darunter sind Bildarchivsysteme (PACS), medizinische Geräte (z. B. Computertomografen) oder darstellende Modalitäten (z. B. Arbeitsstationen zur Befundung) zu verstehen. Der Teil 8 des DICOM-Standards ermöglicht somit die Kommunikation zwischen verschiedenen Systemen zur Verarbeitung medizinischer Bilder.

Shadow, Rishel und Tucker stellen in dem Dokument „Secure HL7 Transactions using Internet Mail“ (Schadow u. a. 1999) eine Empfehlung der Standardisierungsorganisation HL7 für die Übertragung von HL7-v2-konformen Nachrichten mittels verschlüsselter E-Mail vor. Der HL7-v2-Standard ist ein Dokumentenstandard (siehe 2.5.2). Kombiniert mit dem allgemeinen Übertragungsstandard SMTP entsteht daraus ein Protokoll zur Übertragung von standardisiert strukturierten Daten zwischen verschiedenen Kommunikationspartnern.

Der Teil 18 des DICOM-Standards mit dem Titel „Web access to DICOM persistent objects (WADO)“ (NEMA 2009b) beschreibt den Zugriff auf Daten aus DICOM-kompatiblen Bildarchiven mit Hilfe des HTTP-Protokolls. Er bietet somit für den Datenzugriff eine, im Funktionsumfang reduzierte, auf einem verbreiteten Webstandard basierende Alternative zu Teil 8 des DICOM-Standards.

Die beiden letzten Beispiele zeigen die Verwendung allgemeiner Standards zur Übertragung von Daten in Kombination mit medizinischen Dokumentenstandards. Dabei beschreiben die Dokumentenstandards das Format für die übertragenen Nachrichten. Die verwendeten Standards SMTP und HTTP hingegen definieren die Art und Weise der Übertragung sowie die für die Adressierung der Kommunikanten zu verwendenden Mittel. Ein Kommunikant ist hierbei als eindeutig definierte Ressource innerhalb eines ebenfalls eindeutig definierten Systems zu verstehen. Die eindeutige Identifikation kann z. B. durch eine eindeutige Adresse für ein E-Mail-Postfach oder eine URL zu einem serverseitig ausführbaren Skript auf einem Webserver erfolgen.

Alle bisher im Rahmen dieses Kapitels beschriebenen Standards zur Datenübertragung schränken das Format der zu übertragenden Nachricht nicht explizit ein. Soll auf Basis der erhaltenen Nachricht beim Empfänger eine Aktion, die abhängig vom Inhalt der Nachricht ist, initiiert werden, so ist ein dem Sender und Empfänger gemeinsames Verständnis des Inhalts

der Nachricht erforderlich. Das heißt, dem auf dem Server befindlichen Softwarebestandteil müssen das Format und die Struktur der Anfragenachricht bekannt sein.

Grundsätzlich kann man ein Request-Reply-Protokoll als zwei getrennt voneinander zu versendende Nachrichten, die Anfragenachricht und die Antwortnachricht betrachten. Dabei initiiert die erste Nachricht (Anfrage) das Versenden der zweiten Nachricht (Antwort), sodass die Sequenz der beiden Nachrichten eindeutig ist. Damit IT-Systeme mittels Request-Reply-Protokoll in den Rollen Client und Server erfolgreich miteinander kommunizieren können, ist ein auf beiden Seiten eindeutiges Verständnis der Anfrage- und Antwortnachricht nötig. Das System in der Server-Rolle muss die Anfrage des Clients interpretieren können, um in der Lage zu sein, die inhaltlich richtige Antwort zu senden. Die Antwort wiederum muss vom Client in gleicher Weise erwartet werden, wie sie vom Server erzeugt wird. Nur so ist eine richtige Weiterverarbeitung der empfangenen Daten möglich. Mit Hilfe von standardisierten Formaten für zwischen medizinischen Systemen austauschbare Nachrichten kann dieses Problem herstellerunabhängig gelöst werden. Entsprechende Standards werden in Kapitel 2.5.2 vorgestellt.

Im Rahmen des Informationsaustauschs zwischen medizinischen Systemen werden schützenswerte personenbezogene Daten übertragen. Um deren Vertraulichkeit zu wahren, ist deshalb der Einsatz von zuverlässigen Authentifizierungsverfahren, Autorisierungsverfahren und Verschlüsselungstechniken bei der Datenübertragung notwendig. Die Authentifizierung und Verschlüsselung der übertragenen Daten kann dabei auf der Ebene des Übertragungsprotokolls durchgeführt werden. Die Durchführung der Autorisierung bleibt dem eigentlichen Anwendungssystem vorbehalten. Zur Verschlüsselung der übertragenen Nachrichten können verschiedene standardisierte Protokolle zur Anwendung kommen. Sie werden auf Basis der Ebene innerhalb des für die Anwendung gewählten Protokollstapels gruppiert.

Wird eine Verschlüsselung auf der Ebene der Vermittlungsschicht verwendet, so spricht man von Virtual Private Networks (VPN). Diese virtuellen privaten Netzwerke werden auf Basis des IPSec-Protokolls (Atkinson & Kent 1998) realisiert. Es ersetzt dann im Internet-Protokoll-Stack das IP-Protokoll. Die Verschlüsselung auf Ebene des Übertragungsprotokolls ist üblicherweise mittels der Standards Secure-Socket-Layer (SSL) bzw. Transport-Layer-Security (TLS) (Allen & T. Dierks 1999) umgesetzt. Eine Eigenschaft von IPSec und SSL/TLS ist, dass die Verschlüsselung der Daten und gegebenenfalls die Authentifizierung der Kommunikanten außerhalb der eigentlichen Anwendungssoftware stattfindet. Daher kann die Anwendungssoftware nicht gezielt Fragmente von Nachrichten ohne Verschlüsselung übertragen oder die Art der Verschlüsselung kontrollieren. Andererseits ist kein besonderer Aufwand seitens der Anwendungsentwicklung notwendig, um verschlüsselt kommunizieren zu können.

Sollen Nachrichten aber durch eine dritte Instanz, z. B. einem Message-Server, an geeignete Empfänger weitergeleitet werden, so ist durch die Verschlüsselung auf Übertragungsschicht keine zuverlässige Ende-zu-Ende-Verschlüsselung möglich. Die verschlüsselte Übertragung würde in diesem Fall in zwei Abschnitte zerteilt: eine verschlüsselte Übertragung zwischen Sender und Message-Server und eine zweite verschlüsselte Übertragung vom Message-Server

zum Empfänger der Nachricht. Auf dem Message-Server liegt die entsprechende Nachricht während der Übertragung entschlüsselt vor. Soll das vermieden werden, muss auf eine Verschlüsselung der Nachrichteninhalte ausgewichen werden. Bei XML-Nachrichten kann hierfür der Standard XML-Encryption (Imamura u. a. 2002) verwendet werden. Bei E-Mail-Nachrichten kann eine zuverlässige Ende-zu-Ende-Verschlüsselung z. B. mittels S/MIME (Details zu S/MIME siehe IETF 2010; Ramsdell & Turner 2010) erreicht werden.

Neben den beschriebenen Protokollen sind auch andere Mechanismen für die Umsetzung verteilter Krankenakten relevant. Ihnen allen ist gemeinsam, dass sie, wie alle im Kapitel 2.5.1 beschriebenen Standards (außer der DICOM-Netzwerkkommunikation) nicht spezifisch für die Anwendung bei der Umsetzung verteilter Krankenakten sind, sondern als allgemeine IT-Standards domänenunabhängig zum Einsatz kommen.

### 2.5.2. Nachrichten- und Dokumentenstandards

Die Existenz eines gemeinsamen Übertragungswegs zur Übermittlung von Nachrichten zwischen den beteiligten Kommunikanten ist eine notwendige Voraussetzung für deren Kommunikation. Wie bereits am Ende des Kapitels 2.5.1 erläutert, setzt Kommunikation zusätzlich ein einheitliches Verständnis der ausgetauschten Nachrichteninhalte unter den Kommunikationspartnern voraus. Diese Voraussetzung kann durch die Standardisierung des Formats der auszutauschenden Nachrichten erfüllt werden.

Standardisierte Nachrichtenformate gibt es in der Medizin sowohl in der Form älterer, etablierter Standards als auch als neue Standards, die zwar noch keine so weite Verbreitung gefunden haben, dafür aber mit vielen Neuerungen und Verbesserungen aufwarten. Bei den etablierten Standards handelt es sich um den HL7-Nachrichtenstandard in der Version 2, das DICOM-File-Format (NEMA 2009a) der NEMA als Dokumentenstandard sowie die xDT-Nachrichtenstandards (Kassenärztliche Bundesvereinigung 2006) der Kassenärztlichen Bundesvereinigung. Alle drei Standards wurden deutlich vor XML entwickelt und standardisiert. Aus diesem Grund werden jeweils eigene Mechanismen zur Abbildung der Datenstruktur innerhalb der Dokumente verwendet.

**HL7 Version 2**

Die Organisation Health Level Seven International (HL7) ist eine vom American National Standards Institute (ANSI) akkreditierte Standardisierungsorganisation, mit dem Ziel der Entwicklung von Standards für den elektronischen Austausch medizinischer Information. HL7 wurde 1987 gegründet. Der Name Health Level Seven wurde in Anlehnung an den Open Systems Interconnection (OSI) Protokoll-Stack der International Standard Organisation (ISO) gewählt, in dem Anwendungsdienste als 7. Ebene geführt werden (HL7 o. J.).

Bei HL7 Version 2 handelt es sich um einen Nachrichtenstandard, dessen aktuelle Version 2.6 im Dezember 2007 verabschiedet wurde (vgl. HL7 2011). Das Nachrichtenformat von HL7 v2 ist ein eigenständiges textbasiertes Format, in dem das Pipe-Zeichen („|“) zur Trennung und Schachtelung verwendet wird. Seit 2003 ist mit „HL7 Version 2: XML Encoding Syntax, Release 1“ auch ein Standard für die XML-Codierung von HL7-v2-Nachrichten verfügbar.

Die Nachrichten der HL7 Version 2 beschreiben vor allem Ereignisse, die zwischen den verschiedenen Subsystemen innerhalb eines Krankenhauses kommuniziert werden. Für Deutschland relevante Beispiele solcher Nachrichten werden durch die deutschsprachige HL7-Benutzergruppe[7] publiziert. Jede dieser Publikationen enthält für genau einen Anwendungsfall ein exakt beschriebenes und mit Beispielen unterlegtes Nachrichtenformat. Ein Beispiel hierfür ist das Nachrichten-Profil zur Patientenaufnahme (Oemig 2006). Es definiert eine HL7-v2-Nachricht, die bei Aufnahme eines Patienten vom aufnehmenden System an alle registrierten Empfänger gesendet werden soll.

**DICOM File Format**

DICOM ist der Standard für die Verwaltung, Ablage und Verteilung von Bild-Dokumenten, die durch bildgebende Untersuchungsverfahren wie Röntgen, Computertomografie, Magnetresonanztomografie oder auch die Herzkatheter-Untersuchung, Echokardiografie und viele vergleichbare Verfahren entstehen. Für die Speicherung und Übertragung ist in Teil 8 des DICOM-Standards (Media Storage and File Format) (NEMA 2009a) das Format für DICOM-Dateien standardisiert. Es ermöglicht einen vom Hersteller unabhängigen Zugriff auf medizinische Bilder. Für die Darstellung von DICOM-Daten sind auch kostenlose Anzeigeprogramme[8] und Bibliotheken[9] unter Open-Source-Lizenzen verfügbar. Die Standarddokumente zu DICOM sind unter http://medical.nema.org/ frei und kostenlos erhältlich.

Das Format der DICOM-Dateien ist als Binärformat definiert. Es enthält sowohl Metadaten zu den Bildern als auch die Bilder selbst. Eine DICOM-Datei beinhaltet üblicherweise eine zusammengehörige, weil gemeinsam erhobene Serie von Bildern. In den Metadaten einer Serie sind der Patient, der Untersuchende, die Art der Untersuchung, die Zugehörigkeit zu einer Studie sowie Informationen zum Format der Bilder enthalten.

**xDT-Dateiformat**

Unter dem Überbegriff *xDT-Dateiformate* sind die verschiedenen von der Kassenärztlichen Bundesvereinigung (KBV) standardisierten Dateiformate zusammengefasst. Dabei handelt es sich um die Formate:

- ADT (Abrechnungsdatentransfer)
- BDT (Behandlungsdatentransfer)
- GDT (Gerätedatentransfer)
- LDT (Labordatentransfer (Kassenärztliche Bundesvereinigung 2010)).

Seinen Ursprung haben die xDT-Formate in der Entwicklung des ADT-Formats im Jahr 1986 das zum Zweck der elektronischen Abrechnung medizinischer Leistungen bei den Krankenkassen entstanden ist. Das grundlegende xDT-Format ist textbasiert und gemäß der folgenden Struktur aufgebaut.

[7] Webseite der deutschsprachigen HL7 Benutzergruppe: http://www.hl7.de/
[8] Beispiel: ezDICOM, http://sourceforge.net/projects/ezdicom/
[9] Beispiele: http://www.dcm4che.org, http://www.pixelmed.com/ bzw. http://dicom.offis.de/dcmtk.php.de

| Inhalt | Länge | Bedeutung |
|---|---|---|
| Länge | 3 Bytes | Länge des Feldes (Feldinhalt + 9 Bytes für Länge, Kennung und Ende) |
| Kennung | 4 Bytes | Feldkennung, spezifiziert die Art des Feldinhalts |
| Inhalt | Länge – 9 Bytes | Feldinformation |
| Ende | 2 Bytes | CR LF (char 13, 10) |

Tabelle 1: xDT-Formatbeschreibung (http://www.kbv.de/ita/4274.html)

Ein Feld kann in einer xDT-Datei maximal 990 Zeichen enthalten. Wie aus der Spezifikation für den Labordatentransfer LDT (Kassenärztliche Bundesvereinigung 2010) erkennbar ist, handelt es sich bei der Kennung um vierstellige Ziffernkombinationen, die den Inhalt des jeweiligen Feldes spezifizieren. Neben Kennungen für inhaltliche Elemente werden in den spezifischen Formatdefinitionen (LDT usw.) auch Kennungen für Strukturierungselemente wie z. B. einen Header-Bereich definiert. So legt bei LDT die Feldkennung 8000 beispielsweise die Satzart des nächsten Abschnitts innerhalb einer Datei fest und trennt somit die Abschnitte. Innerhalb eines Satzes der Satzart 8220, dem Labor-Datenpaket-Header, dient die Feldkennung 0212 als Kennung für das Feld mit der lebenslangen Arztnummer (Kassenärztliche Bundesvereinigung 2010:S. 21).

Die xDT-Standards wurden zum Zeitpunkt ihrer Entwicklung für die Übertragung von Daten zwischen verschiedenen Praxissystemen mittels Datenträgern wie z. B. Disketten geplant.

Die drei bisher genauer beschriebenen Standards sind primär für die Kommunikation innerhalb von Institutionen oder für den Austausch von Daten mittels Datenträgern konzipiert. Aus diesem Grund sind verschiedene, für sichere verteilte Nutzung relevante Aspekte wie Sicherheit und direkte Integrationsfähigkeit nicht berücksichtigt. DICOM ist bezogen auf den Dateistandard auch im Rahmen von einrichtungsübergreifenden Szenarien gut verwendbar. Probleme bereitet hier der zugehörige Übertragungsstandard. Er beinhaltet keine Mechanismen zur Authentifizierung und zur Vergabe von Berechtigungen für den Zugriff oder die Modifikation von Objekten. Außerdem unterstützt er keine verschlüsselte Kommunikation auf Transport- oder Nachrichtenebene. Die üblicherweise in DICOM-fähigen PACS-Systemen verfügbaren Sicherheitsmechanismen beschränken sich auf Definition von Zugriffsrechten auf Basis von IP-Adressen.

Bei HL7 v2 und xDT werden die vorhandenen Krankenhaus- oder Arztpraxisinformations-systeme als abgeschlossene Einheiten betrachtet. Sie werden durch den Datenaustausch lediglich um zusätzliche Daten angereichert oder mit anderen Systemen synchronisiert. Dieses Vorgehen setzt eine hohe Gleichheit bei den Datenbeständen voraus und führt zu einer großen Menge an zu versendenden Nachrichten. Mit zunehmender Zahl der Systemknoten steigt die Anzahl der zu versendenden Nachrichten überproportional an. Für n Systemknoten gilt eine maximale Zahl von (n * n-1)/2 notwendigen Nachrichten. Bei modernen verteilten Versorgungsszenarien ist häufig eine andere, weniger kommunikationsintensive und stärker am Verlauf des Versorgungsprozesses orientierte Form der kommunizierten Dokumentation

nötig. Außerdem weisen die beschriebenen Formate Defizite bezüglich ihrer flexiblen Anpassbarkeit an sich stetig ändernde verteilte Versorgungspfade auf.

Aus diesem Grund wurden in den letzten Jahren von verschiedenen Standardisierungsorganisationen und Interessengruppen neue XML-basierte Dokumentenstandards entwickelt.

**HL7 Version 3 Nachrichten**

HL7 Version 3 ist der XML-basierte Nachfolgestandard von HL7 Version 2. Seine Nachrichtentypen lassen sich über einen definierten mehrstufigen Prozess vom HL7 Reference Information Model (RIM) ableiten (Haas 2006:S. 330).

**HL7 Clinical Document Architecture (CDA)**

HL7 CDA ist ein XML-basierter Standard für den Austausch und die Speicherung medizinischer Dokumente (Blobel 2008:S. 62). CDA-Dokumente können beispielsweise Überweisungsdokumente, die Behandlungsdokumentation oder OP-Berichte abbilden. Die CDA definiert dabei, ausgehend vom HL7-Version-3-Nachrichtenstandard und dem HL 7 RIM ein XML-Schema für beliebige medizinische Dokumente.

CDA-Dokumente sind in die Bereiche CDA-Header und CDA-Body unterteilt. Der CDA-Header wird im Gegensatz zum CDA-Body nicht von einem eigenen XML-Element umfasst. Er beinhaltet laut Spezifikation die Metadaten zu dem Dokument, das im CDA-Body enthalten ist. Im Header sind somit Informationen zum Dokument selbst, den an der Erstellung des Dokuments beteiligten Personen und zu der Person, die der Inhalt des Dokuments betrifft, also dem Patienten enthalten (Haas 2006:S. 331–343). Der CDA-Body wird durch das XML-Element StructuredBody gekennzeichnet. Er kann sowohl strukturierte Informationen in Form weiterer eingeschachtelter XML-Elemente, Verweise auf andere Dokumente, als auch unstrukturierte Informationen in Form von Binärdaten beinhalten. Ein Ausschnitt aus der XML-Schema-Beschreibung von CDA befindet sich in Form des Elements StructuredBody im Anhang C dieser Arbeit.

Konkrete CDA-Dokumente werden in Form von CDA Clinical Templates vorgeschlagen bzw. standardisiert. Sie können ausgehend vom Template auf Basis eines von HL7 spezifizierten Modellierungsprozesses angepasst werden, ohne die Kompatibilität zum ursprünglichen Template zu verlieren. Die zugrunde liegende Idee ist eine logische Weiterentwicklung des Konzepts der HL7-v2-Nachrichtenprofile. CDA Clinical Templates gehen allerdings deutlich über dieses Konzept hinaus und können auch Beschreibungen über andere klinische Konzepte und Abläufe umfassen (Blobel 2008:S. 63).

**Continuity of Care Record (CCR)**

Der Continuity of Care Record wird bei der American Society for Testing and Materials (ASTM) als Standard E2369 geführt. Er definiert die Dokumentenstruktur einer Krankenakte in Form eines XML-Dokuments das sowohl administrative- als auch klinische Informationen zu je einem Patienten enthält (Blobel 2008:S. 64). Zielsetzung des Standards ist es, die direkte Weiterführung der Versorgung durch nachfolgende Leistungsstellen in verteilten

Versorgungsszenarien zu unterstützen. Dabei kann ein Continuity of Care Record technisch sowohl eine partielle als auch eine umfassende Krankenakte beinhalten.

Waegemann beschreibt die Einsatzszenarien für den CCR als sehr vielfältig (C. P. Waegemann 2007:S. 19). Sie umfassen die Weitergabe von Daten bei der Überweisung von Patienten, den allgemeinen Transfer behandlungsrelevanter Daten zwischen medizinischen Institutionen, die Rückübermittlung von Daten im Falle einer Entlassung oder auch für die Erstellung persönlicher Gesundheitsakten.

In den USA wird der CCR-Standard inzwischen von einer Vielzahl der Hersteller von Krankenhaus- und Praxisinformationssystemen implementiert (Demski u. a. 2008). Auch die Gesundheitsakte „Google Health“ des Anbieters Google verwendete CCR als Basisstandard für die Datenstrukturen seiner Akten (Google o. J.). Der Dienst Google Health wurde zum 01.01.2013 endgültig abgeschaltet und ist als ein gescheiterter Versuch einer persönlichen Gesundheitsakte zu betrachten (A. Brown & Weihl 2011).

**Continuity of Care Document (CCD)**

Das Continuity of Care Document basiert auf der Idee, die Vorteile von CDA und CCR zu kombinieren. Es ist wie der Continuity of Care Record ein XML-Dokument für die fortlaufende Behandlungsdokumentation, wobei das CCD auf dem XML-Schema der CDA aufbaut, während CCR eine nicht zur CDA kompatible XML-Struktur besitzt (HL7 2010). Folglich handelt es sich beim CCD um ein spezielles CDA-Template, das entsprechend der HL7-Vorgaben für die Entwicklung von CDA-Templates und der fachlichen Anforderungen des CCR entwickelt worden ist.

Der CCD-Standard ist jünger als CCR und CDA. Er ist aus diesem Grund derzeit noch nicht so stark verbreitet wie CCR. Der Gesundheitsakten-Dienst „Microsoft Health Vault“ unterstützt allerdings bereits CCR und CCD als Standards für den Import und Export von Patientendaten (Microsoft o. J.).

### *2.5.3. Semantikstandards*

Standardisierte Kommunikations- und Übertragungswege ermöglichen in Kombination mit ebenfalls standardisierten Nachrichten- und Dokumentenformaten eine Übertragung von zuverlässig lesbaren Daten zwischen verschiedenen Kommunikanten. Sie ermöglichen die automatisierte Ermittlung von elementaren Bestandteilen komplexer übermittelter Objekte. Die Verwendbarkeit der Werte dieser Elemente ist jedoch an eine weitere Bedingung geknüpft: ein einheitliches Verständnis für die Bedeutung der übermittelten Werte. Bei numerischen Messgrößen kann dieses einheitliche Verständnis z. B. durch die Angabe der Einheit der Messgröße erreicht werden. In anderen Fällen wird die Eindeutigkeit durch genormte Aufzählungen valider Werte gewährleistet. Diese genormten Aufzählungen werden Semantikstandards genannt. Für den Bereich der medizinischen Dokumentation ist eine Vielzahl von Semantikstandards verfügbar. Zur Veranschaulichung ihrer Zielsetzung werden die drei Standards ICD-10, SNOMED CT und LOINC erläutert.

**ICD-10**

Die International Classification for Diseases (ICD) ist ein Standard der Weltgesundheitsorganisation (WHO) zur semantisch eindeutigen und sprachunabhängigen Angabe von diagnostizierten Krankheiten. Die zehnte Ausgabe der ICD ist seit 1994 in den Mitgliedstaaten der WHO im Gebrauch (WHO o. J.). Mit der Nutzung von ICD-Codes wird das Ziel verfolgt, eine international einheitliche Klassifikation von Erkrankungen zu verwenden. Diese wiederum ermöglicht es statistische Auswertungen etwa bezüglich der Verbreitung und Häufigkeit von Erkrankungen zu führen (WHO 2004:S. 8). Die Verwendung von eindeutigen Codes anstelle natürlichsprachlicher Bezeichnungen für Erkrankungen verbessert neben den statistischen Möglichkeiten der WHO auch jegliche weitere automatisierte Auswertung und Verarbeitung entsprechend codierter Patientendaten.
Die Geschichte der ICD reicht bis 1850 zurück. Ursprünglich wurde das Codierungssystem dazu entwickelt, verschiedene Todesursachen zu kategorisieren, um deren Häufigkeit systematisch ermitteln zu können (WHO o. J.).

Die WHO gliedert die Standarddokumente zu ICD-10 in drei Bände, Band 1 behandelt die Klassifikation, Band 2 bietet einen Benutzerleitfaden für die Durchführung der Klassifikation und Band 3 eine alphabetische Auflistung zur Klassifikation (WHO 2004:S. 17). Die ICD-Klassifikation selbst ist in 21 Kapitel unterteilt. Jeder ICD-Code beginnt mit einem Buchstaben, dieser ist genau einem Kapitel zugeordnet, wobei ein Kapitel auch mehr als einen Buchstaben besitzen kann. Der Buchstabe H stellt dabei eine Ausnahme dar, denn er ist sowohl dem Kapitel 7 als auch dem Kapitel 8 zugeordnet. Die Kapitel 1 bis 17 beinhalten Krankheiten und andere krankhafte Zustände, das Kapitel 18 beinhaltet Symptome. Verletzungen und Vergiftungen werden in Kapitel 21 definiert (WHO 2004:S. 18). Die Kapitel selbst sind in homogene Gruppen (Blocks), bestehend aus mehreren logisch zusammengehörigen dreistelligen ICD-Codes gegliedert (WHO 2004:S. 18, 2.4.3 Blocks of categories). Das Format dreistelliger ICD-Codes besteht aus einen Buchstaben, gefolgt von zwei Ziffern. Die meisten dreistelligen ICD-Codes sind weiter in vierstellige ICD-Codes untergliedert (WHO 2004:S. 18, 2.4.5 Four-character subcategories). Dabei umfasst ein vierstelliger ICD-Code fünf Zeichen, da das letzte Zeichen durch einen Punkt abgetrennt wird.

Die Systematik der ICD-10-Codes lässt sich anhand eines Beispiels erläutern. Diesem liegt eine histologisch bestimmte Tuberkulose mit dem ICD-Code A15.2 zugrunde. Die Datenbasis zur Ermittlung des ICD-Codes für Tuberkulose ist der ICD-10-Onlinedienst[10] der WHO. Betrachtet man die einzelnen Kapitel der ICD, so kann Tuberkulose prinzipiell in zwei der Kapitel zu suchen sein: entweder im Kapitel 10 über die Krankheiten des Atmungssystems *(„Diseases of the respiratory system")* oder in Kapitel 1 über besondere ansteckende Krankheiten *(„Certain infectious and parasitic diseases")*. In Kapitel 1 befindet sich die Gruppe A15-A19, Tuberkulose, die den Code A15 *"Respiratory tuberculosis, bacteriologically and histologically confirmed"* beinhaltet. Dass die Diagnose auf der Basis eines histologischen Befunds erfolgt ist, kann durch den vierstelligen Code A15.2 zum Ausdruck gebracht werden.

[10] http://apps.who.int/classifications/apps/icd/icd10online/

**SNOMED CT**

Die Systematized Nomenclature of Medicine - Clinical Terms (SNOMED CT) ist eine ursprünglich vom College of American Pathologists entwickelter umfassender Semantikstandard für die Codierung klinischer Begriffe (Concepts) (U.S. National Library of Medicine 2009). Die Nomenklatur von SNOMED CT enthält derzeit über 311.000 klinische Konzepte oder Begriffe (IHTSDO o. J.). Das Codierungssystem von SNOMED CT ist deutlich umfangreicher als das der ICD-10-Codes.

SNOMED CT ist das Ergebnis der Zusammenführung von SNOMED RT des College of American Pathologists und den Clinical Terms Version 3 (CTV3) des britischen National Health Service (NHS) (IHTSDO 2009:S. 8). Die Terminologie besteht aus Begriffen, Termen und Beziehungen. Sie verfolgt das Ziel, beliebige klinische Informationen eindeutig abbilden zu können. Dazu ist der Inhalt von SNOMED CT in 19 Hierarchien gegliedert.

- Clinical finding
- Physical force
- Procedure
- Event
- Observable entity
- Environment or geographical location
- Body structure
- Social context
- Organism
- Situation with explicit context
- Substance
- Staging and scales
- Pharmaceutical/biologic product
- Linkage concept
- Specimen
- Qualifier value
- Special concept
- Record artifact
- Physical object

Tabelle 2: Hierarchien von SNOMED CT (IHTSDO 2009:S. 8)

Im Sinne von SNOMED CT ist ein Begriff (Concept) ein klinischer Begriff einschließlich seines numerischen Codes, der ihn eindeutig identifiziert (IHTSDO 2009:S. 9). Dieser Code wird ConceptID genannt. Es gibt allgemeine Begriffe und spezielle Begriffe. Die Granularität von Begriffen ist folglich variabel. Die ConceptID besitzt keine interne Struktur, die hierarchische oder sonstige Gliederungsprinzipien abbildet. Begriffe werden durch die in ihnen enthaltenen Terme beschrieben. Terme werden durch DescriptionIDs identifiziert (IHTSDO 2009:S. 10).

Jeder Begriff hat einen eindeutigen „Fully Specified Name (FSN)“, der es ermöglicht, ihn unmissverständlich zu benennen (IHTSDO 2009:S. 10). Er besteht aus einem natürlichsprachlichen Bezeichner und einem darauf folgenden semantischen Tag, der die semantische Kategorie des Begriffs bestimmt.

Zusammenfassend betrachtet, beinhaltet ein Begriff in SNOMED CT folglich einen bevorzugten textuellen Bezeichner für eine klinische Tatsache, einschließlich der zugehörigen Synonyme. Diese Kombination ist durch die vergebene Nummer eindeutig identifizierbar. Außerdem ist der Begriff durch den semantischen Tag im FSN einer Klasse von Begriffen (Krankheit, Verletzung, Einflussfaktor usw.) zugeordnet.

Das folgende Beispiel zeigt den Aufbau von SNOMED CT (Quelle: IHTSDO 2009:S. 10).

ConceptID 22298006:

- Fully Specified Name: *Myocardial infarction (disorder)*
  DescriptionID 751689013
- Preferred term: Myocardial infarction
  DescriptionID 37436014
- Synonym: Cardiac infarction
  DescriptionID 37442013
- Synonym: Heart attack
  DescriptionID 37443015
- Synonym: Infarction of heart
  DescriptionID 37441018

**LOINC**

Neben SNOMED CT gibt es mit Logical Observation Identifiers Names and Codes (LOINC) einen weiteren sehr umfangreichen Semantikstandard für die Medizin. Die Wurzeln dieses Standards liegen in der Labormedizin. Die Entwicklung von LOINC wurde 1994 am Regenstrief Institut begonnen (D. Vreeman 2010). Seit April 1996 ist LOINC im Internet öffentlich zugänglich. Seither wurden 13 überarbeitete Versionen veröffentlicht. Derzeit beinhaltet LOINC mehr als 30.000 eindeutige untersuchungsbezogene Begriffe (McDonald u. a. 2010:S. 4). Im Gegensatz zu SNOMED CT ist die Verwendung von LOINC nicht kostenpflichtig. LOINC-Codes sind sowohl für Laborergebnisse als auch für klinische Ergebnisse verfügbar. Im Bereich der klinischen Ergebnisse wird sowohl die Human- als auch die Veterinärmedizin berücksichtigt (McDonald u. a. 2010:S. 6–7).

Innerhalb von LOINC werden die Begriffe (Namen der Untersuchungen) durch die folgende Systematik beschrieben:

<Analysegegenstand>:<Art der Untersuchung bzw. Messung>:<Zeitaspekt>:<Quellsystem der Probe>:<Einheit der Messung>:<Messmethode>

Jeder dieser Begriffe erhält einen eindeutigen LOINC-Code zugeordnet. Dabei haben LOINC-Codes keine definierte innere Struktur zur Abbildung von Zusammenhängen. Lediglich die letzte Stelle des LOINC-Codes enthält ein Prüfzeichen zur Erkennung von Fehlern in den Codes (McDonald u. a. 2010:S. 7).

Beispiele für LOINC-Codes und Begriffe:

| LOINC-Code | LOINC-Name |
|---|---|
| 12907-2 | Sodium:SCnc:Pt:RBC:Qn<br>Erklärung: (Natrium:Substanzkonzentration:Messung zu Zeitpunkt:in roten Blutkörperchen:Quantitativ) |
| 13895-8 | Sodium:SCnc:Pt:Milk:Qn<br>Erklärung: (Natriumkonzentration in Milch, Zeitpunktmessung) |

Tabelle 3: Beispiele für LOINC-Codes

Neben dem LOINC-Namen werden in der LOINC-Datenbank im Feld RelatedNames2 Synonyme natürlichsprachliche Bezeichner für den LOINC-Begriff aufgelistet. Der LOINC-Code eignet sich für die Verwendung in Datenfeldern von Nachrichten da er, als Feldbezeichner verwendet, die Bedeutung des Feldinhalts eindeutig spezifiziert.

### 2.5.4. Referenzmodelle und Architekturstandards

Neben den verschiedenen in den Kapiteln 2.5.1 bis 2.5.3 beschriebenen Kommunikations-, Nachrichten-, Dokumenten- und Semantikstandards beeinflussen derzeit drei Architekturstandards und ein Referenzmodell die Entwicklung elektronischer Kranken- und Patientenakten. Referenzmodellen und Architekturstandards ist es gemeinsam, dass sie dazu dienen, auch im Bereich der Interna der an verteilten Krankenakten beteiligten Systeme Gemeinsamkeiten zu fördern, um so die Erreichung möglichst vollständiger Interoperabilität zu erleichtern.

**HL7 RIM**

Das HL7 Reference Information Model (RIM) ist ein generisches Daten- und Informationsmodell, mit dem sich beliebige Handlungen im Rahmen einer rollenbasierten Beziehung zwischen Individuen abbilden und somit dokumentieren lassen. Es besteht aus einer Menge von sechs Basisklassen, die durch Vererbung verfeinert werden können. Für die Basisklassen Entity, Role und Act sind im Referenzmodell bereits eine Vielzahl definierter Subklassen vorgesehen.

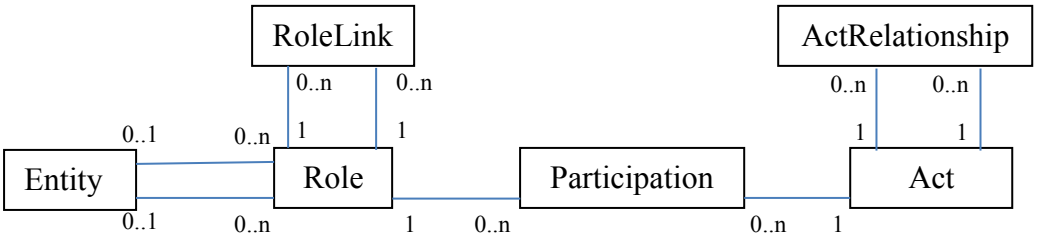



Abbildung 21: Die Basisklassen des HL7 RIM (vgl. Haas 2006:S. 360)

Das HL7 RIM wird als Grundlage für die Entwicklung konkreter HL7-Nachrichten oder HL7-CDA-Dokumente verwendet. Dazu existiert ein definierter Prozess (übersichtlich dargestellt in Kaspar 2005:S. 18), in dem mittels mehrerer Schritte, ausgehend vom RIM, über ein sogenanntes Domain-Message Information Model (D-MIM) und ein daraus abgeleitetes Refined-Message Information Model (R-MIM) eine Hierarchical Message Description (HMD) erarbeitet wird. Die HMD stellt dabei eine technologieunabhängige Beschreibung des logischen Aufbaus einer Nachricht dar. Soll die so entwickelte Nachricht als HL7 Version 3 Nachricht eingesetzt werden, muss sie mit Hilfe einer passenden Implementation Technology Specification (ITS) in ein konkretes Format wie z. B. XML und XSD umgesetzt werden.

Über die allgemeine Verwendbarkeit des HL7 RIM herrschen in der Literatur unterschiedliche Meinungen. Neben umfangreicher Zustimmung (Bointner & Duftschmied 2008; Sanchez u. a. 2008; Eggebraaten u. a. 2007) wird die Allgemeingültigkeit von verschiedenen Autoren etwa aufgrund von Mehrdeutigkeiten in den Definitionen des

Standarddokuments und anderen Problemen bezweifelt (B. Smith & Ceusters 2006; Schadow u. a. 2006).

**GEHR**

Good European Health Record (GEHR) ist ein europäischer Standardisierungsversuch für eine Akten-Architektur und für Lösungsansätze zur Erreichung von Interoperabilität zwischen elektronischen Krankenakten. Seine Ursprünge gehen auf die Jahre 1991 bis 1995 zurück (Haas 2006:S. 366). Die Ergebnisse des Good European Health Record sind von 1997 bis 2001 (Beale u. a. 2007:S. 15) im Rahmen des australischen Projekts Good Electronic Health Record (GeHR) weiterentwickelt worden (openEHR 2008; Ingram 2002).

Die GEHR-Architektur enthält unter anderem ein eigenes Informationsmodell, das die Minimalanforderungen für Struktur und Inhalt eines GEHR-kompatiblen Systemknotens spezifiziert. Außerdem beinhaltet die Architektur ein spezifisches Format für den Datenaustausch zwischen heterogenen Systemknoten (David Lloyd 1995:S. 9). Das Informationsmodell ist wiederum in das GEHR Object Model und das Metamodell für die Archetypen aufgeteilt (Haas 2006:S. 366). Archetypen sind Datenmodell-Muster für die Abbildung klinischer Begriffe (clinical Concepts).

Bei GEHR ist das zentrale Ordnungskonzept innerhalb einer Akte die Transaktion. Sie umfasst ein oder mehrere Dokumente, die zu genau einem Besuch des Patienten beim erfassenden Gesundheitsdienstleister entstanden sind. Für eine Transaktion sind der Zeitpunkt der Transaktion, die medizinische Einrichtung, in der die Transaktion durchgeführt wurde, und der verantwortliche Mitarbeiter eindeutig definiert. Sie bildet eine abgeschlossene Einheit, die nachträglich nicht mehr geändert werden kann. Notwendige Änderungen werden durch das Hinzufügen neuer Versionen eingebracht.

Transaktionen werden in die Transaktionstypen

- Verwaltungstransaktion (Administration),
- Transaktion bei Patientenkontakt (Contact),
- Transaktion mit zusammenfassendem Inhalt (Summary),
- Transaktionen, die Folgetransaktionen auslösen (Trigger),
- Informationen mit externer Quelle (Report),
- Transaktionen mit für die Weiterbehandlung relevantem Inhalt (Continuing Care) sowie
- Transaktionen, deren Inhalt bei jedem Öffnen der Akte zuerst angezeigt werden sollen (Nota Bene),

eingeordnet (David Lloyd 1995:S. 34–36). Im Rahmen der GEHR-Projekte wurde neben einer umfangreichen Zahl an Dokumenten auch lauffähige Software z. B. in der Programmiersprache Eiffel entwickelt.

**openEHR**

openEHR ist ein Projekt der openEHR Foundation, einer Non-Profit-Organisation des University College London und der Ocean Informatics Pty. Ltd. in Australien. Es baut auf dem europäischen und dem australischen GEHR-Projekt auf. Dabei wird versucht, die Ansätze von HL7, CEN13606 und GEHR zu berücksichtigen und in einer gemeinsamen Patientenaktenarchitektur zusammenzuführen (Haas 2006:S. 366; Beale & Sam Heard 2008:S. 10 ff.).

Das openEHR Projekt verfolgt einen generischen Modellierungsansatz für die Erstellung der Bestandteile elektronischer Patientenakten. Wie GEHR verwendet auch openEHR dazu das Konzept der Archetypen. Die Archetypen des openEHR Archetype Object Models werden mittels einer eigenen formalen Sprache, der Archetype Definition Language (ADL) (Beale & Sam Heard 2007) beschrieben und definiert. Dadurch können die definierten Archetypen automatisiert weiterverwendet werden. Alle Spezifikationen zu openEHR sind unter http://www.openehr.org öffentlich zugänglich.

Das Projekt gliedert seine Aktivitäten in technische- und klinische Belange. Die technischen Aktivitäten des openEHR-Projekts teilen sich in ein Projekt zur Ermittlung und Analyse der Anforderungen, ein Projekt zur Entwicklung der Architektur, ein Projekt zur Umsetzung einer prototypischen Implementierung, ein Projekt zur Koordination des Projekts mit verschiedenen Standardisierungsorganisationen und ein Projekt zur Vermittlung von Wissen über openEHR an Dritte. Die klinischen Belange des openEHR-Projekts werden in die Teilprojekte Begriffe und Terminologien, Archetypen und Templates, Standards sowie Lehre gegliedert (Beale u. a. 2006:S. 7–9).

Ziel des Projekts ist neben der Erstellung normativer Dokumente auch die Entwicklung eines unter einer OpenSource-Lizenz frei verfügbaren Demonstrationssystems für die openEHR-Architektur.

**ISO EN 13606**

Ein weiterer Architekturstandard ist der CEN-Standard 13606 (Electronic Healthcare Record Communication), der inzwischen auch bei der internationalen Standardisierungsorganisation ISO als Standard EN 13606-1 bis 13606-5 verabschiedet und veröffentlicht ist. Wie openEHR verfolgt auch dieser Standard einen stark generischen, auf Archetypen basierenden Ansatz für die Definition der Datenmodelle von Krankenakten (Haas 2006:S. 363). Der Standard befasst sich vorwiegend mit der Abbildung der Inhalte und Strukturen von Krankenakten sowie mit den Themen Sicherheit und den Inhalten auszutauschender Nachrichten.

Die vorgestellten Referenzmodelle und Architekturstandards befassen sich alle mit den Fragen der Abbildung von Inhalten elektronischer Patientenakten und einer darauf aufbauenden Umsetzung dieser Inhalte in Nachrichten zum Austausch von Information zwischen verschiedenen Systemen. Der Begriff Architekturstandard kann dabei zu Missverständnissen führen, da Architekturstandards hier nicht den Anspruch erheben, die System- oder Softwarearchitektur des der elektronischen Krankenakte zugrunde liegenden IT-Systems zu

standardisieren. Stattdessen befassen sich die genannten Standards mit Strukturkonzepten zur eindeutigen Abbildung der Akte, sozusagen der Architektur einer Patientenakte.

Hierbei verändern die genannten Referenzmodelle und Architekturstandards die Ablageform medizinischer Information von unstrukturierten Texten oder proprietär strukturierten Formaten hin zu verschiedenen standardisierten, semantisch eindeutigen Abbildungsformen strukturierter Dokumente. Diese Dokumente sind der Inhalt von zwischen verschiedenen Institutionen interoperablen elektronischen Akten.

Aktuell in der Praxis eingesetzte medizinische Informationssysteme sind nicht immer dazu geeignet, derart feingranular strukturierte Daten zu speichern. Oft findet sich eine aus natürlichsprachlichem Fließtext und einigen wenigen zusätzlichen semantischen Tags, wie z. B. ICD-10 Codes, zusammengestellte Abbildungsform. Die im natürlichsprachlichen Text enthaltene Semantik kann nicht zuverlässig automatisch durch Software erschlossen und in eindeutige Tags umgesetzt werden. Das macht eine automatisierte Migration von alten eigenen auf neue standardisierte Dokumenten- und Aktenformate unmöglich. Aus diesem Grund kann eine Migration nur schrittweise und unter Beibehaltung der Verwendbarkeit alter Dokumente durchgeführt werden. Voraussetzung dafür ist die Verfügbarkeit entsprechend erweiterter neuer Versionen des in einer Organisation bereits verwendeten Informationssystems.

### 2.5.5. Schwerpunkte der Standardisierung

Die Kapitel 2.5.1 - 2.5.4 stellen eine Auswahl der in Literatur oder Praxis verbreiteten Standards aus dem thematischen Bereich verteilter Krankenakten vor. Dabei lässt sich eine allen beschriebenen Standards gemeinsame Eigenschaft erkennen. Der zentrale Fokus der Standards liegt auf der Abbildung medizinischer Inhalte und der inneren Struktur der Akte. Einige der Standards behandeln auch konkrete Mechanismen zur Nachrichtenübertragung oder definieren abstrakte Operationen für übliche aktenbezogene Arbeitsschritte. Bei der Abbildung der Inhalte reicht die Spanne von der Definition von Strukturen für Nachrichten über die von Dokumenten bis hin zu ganzen Patientenakten. Zur Erreichung semantischer Interoperabilität wird kontrolliertes Vokabular definiert, das in Nachrichten, Dokumenten und Akten eingesetzt werden kann. Besonders bei den neuen Nachrichten-, Dokumenten- und Architekturstandards ist eine explizite Integration von Semantikstandards bereits durch die jeweiligen Standards vorgesehen. Folglich liegt einer der Schwerpunkte der Standardisierung auf Inhalt und Struktur der Patientenakten.

Das mehrfach verwendete Konzept der Archetypen für klinische Begriffe (clinical Concepts) ist ein Ansatz zur Erstellung wiederverwendbarer Muster bei der Datenmodellierung. Diese Konzepte der Standards dienen der Erreichung von inhaltlicher Erweiterbarkeit, Veränderbarkeit und Anpassbarkeit.

Ein weiterer Aspekt, der in einigen, aber nicht allen der Standards erwähnt wird, ist die Sicherheit. Er nimmt aber im Vergleich zu den inhaltlichen Aspekten bei der Standardisierung einen, bezogen auf den Umfang der Ausführungen, eher untergeordneten Stellenwert ein.

Aufgrund der vorangegangenen Analyse der verschiedenen für verteilte Krankenakten besonders bedeutsamen Standards sind folgende für verteilte Krankenakten relevanten Schwerpunkte zu identifizieren:

- Inhalt,
- Struktur,
- Sicherheit,
- Erweiterbarkeit,
- Veränderbarkeit und
- Anpassbarkeit.

Die Schwerpunkte Inhalt und Struktur sind dabei neu ermittelt: die Schwerpunkte Sicherheit und Erweiterbarkeit wurden bereits am Ende von Kapitel 2.4.2 durch die Analyse der nichtfunktionalen Anforderungen bestimmt.

## 2.6. Projekte aus Forschung und Praxis

Eine vollständige Betrachtung aller Forschungsprojekte zu verteilten Krankenakten ist aufgrund ihrer großer Anzahl nicht möglich. Die Klassifikation einer Auswahl exemplarischer Projekte anhand definierter Kriterien ist jedoch geeignet, um die unterschiedlichen Konzepte realisierter oder geplanter verteilter Krankenaktensysteme zu beleuchten. Außerdem wird auf diese Weise die Vielfalt der Kombinationen aus den verwendeten Architekturkonzepten, Kommunikationsparadigmen, Standards, Produkten und Technologien sichtbar.

Eine explizite Unterscheidung zwischen Projekten der Forschung und der Praxis wird bei der nachfolgenden Betrachtung nicht getroffen. Für die Ermittlung des Ergebnisses ist eine solche Unterscheidung, auch wegen der in den Projekten häufig nicht eindeutig definierten Grenze zwischen Forschung und Praxis, nicht relevant.

### 2.6.1. Klassifikations- und Unterscheidungskriterien

Für die Durchführung einer systematischen Betrachtung von Projekten aus Forschung und Praxis ist die vorherige Festlegung von Kriterien zur Unterscheidung der spezifischen Eigenschaften der dabei entstandenen Systeme notwendig. Die Auswahl der Kriterien basiert sowohl auf den bereits in den bisherigen Kapiteln identifizierten Schwerpunkten als auch auf Erkenntnissen aus einer vorangegangenen, ohne explizite Systematik durchgeführten Literaturrecherche zu Beispielprojekten. Um die Zuordnung eindeutig zu gestalten wird zusätzlich für jedes Kriterium ein kontrolliertes Vokabular für die möglichen Ausprägungen definiert.

Die Auswahl der Kriterien erfolgt im Wesentlichen auf Basis der folgenden Fragen:

- Wodurch lassen sich IT-Systeme gleicher Gattung unterscheiden?
- Was beeinflusst die spätere Benutzbarkeit des Systems?
- Was beeinflusst die möglichen Nutzungsarten des Systems?
- Was beeinflusst die Interoperabilität des Systems?
- Was beeinflusst die Zuverlässigkeit des Systems?
- Was beeinflusst die Flexibilität des Systems?
- Was beeinflusst die Sicherheit des Systems?

Daraus ergeben sich die folgenden Klassifizierungskriterien:

- Verbreitungsgebiet der Krankenakte
- Menge der Knoten mit eigener Datenspeicherung
- Redundanz
- Art der verfügbaren Dokumente und Inhalte
- Inhaltlicher Umfang der Akte
- Dominierende Muster bzw. Prinzipien der Softwarearchitektur
- Kommunikationsform
- Rollen der Kommunikanten
- Art der Erweiterbarkeit und Veränderbarkeit

- Kontext der Erweiterbarkeit und Veränderbarkeit
- Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts
- Art der Integration in die bestehende System- und Prozesslandschaft.

Zu den Klassifikationskriterien ergeben sich jeweils spezifische Ausprägungen. Bei manchen der Kriterien ist eine vollständige Auflistung aller möglichen Ausprägungen nicht zuverlässig erreichbar. In diesen Fällen wurden einige typische Vertreter ausgewählt. Die eindeutige Definition der Ausprägungen ermöglicht eine genaue Einordnung konkreter Systeme und fördert so deren Vergleichbarkeit. Außerdem wird dadurch ein Überblick über die in real existierenden Systemen verwendete Bandbreite der Systemgestaltung möglich.

Verbreitungsgebiet der Krankenakte

Um die Größe und den Umfang eines Systems zur Verwaltung verteilter Krankenakten zu bewerten, sind verschiedene Kriterien relevant. Eines dieser Kriterien ist die Größe des geografischen Gebietes, in dem das System verwendet wird. Die Betrachtung dieses Kriteriums ist allerdings ohne die zusätzliche Einschränkung durch die Berücksichtigung der Verbreitung und Verteilung der Nutzer des Systems innerhalb des Gebiets ungenau und kann zu falschen Annahmen führen. Aus diesem Grund wird die Verbreitung des Systems durch die folgenden Ausprägungen spezifiziert:

- Lokal: innerhalb einer Einrichtung
- Verbund: lokal abgegrenzter Verbund von Einrichtungen
- Regionaler Verbund: regional abgegrenzter Verbund von Einrichtungen
- Regional: alle Einrichtungen einer Region
- National: alle Einrichtungen eines Landes
- International: alle Einrichtungen mehrerer beteiligter Länder.

Menge der Knoten mit eigener Datenspeicherung

Die Anzahl der Knoten in einem System, die Daten selbst speichern und anderen Systemteilnehmern zur Verfügung stellen, kann sowohl über die Architektur des Systems als auch, abhängig von dessen Architektur, über die Größe des Systems Auskunft geben. Eine exakte Anzahl ist aus den Veröffentlichungen zu den Systemen nicht immer zu ermitteln. Aus diesem Grund werden für die Anzahl der Knoten mit eigener Datenspeicherung die folgenden drei Ausprägungen festgelegt:

- Zentrale Datenspeicherung: Ein zentraler Knoten speichert die Daten
- Dezentrale Server: Mehrere spezialisierte Knoten speichern die Daten
- Vollständig dezentrale Datenspeicherung: Jeder Knoten speichert seine Daten selbst.

Redundanz

Die Frage, ob das gleiche Dokument in der verteilten Krankenakte an mehreren Stellen gespeichert wird, kommt durch das Kriterium Redundanz zum Ausdruck. Der Begriff *Redundanz* beschreibt allgemein das eigentlich aus der Sicht des Informationswerts des

Objekts überflüssige mehrfache Vorhandensein eines Objektes. Bezüglich der Ausfallsicherheit und der Datensicherung kann die Redundanz allerdings durchaus positive Auswirkungen haben oder notwendig sein. Zur Betrachtung des Kriteriums *Redundanz* werden die folgenden Ausprägungen verwendet:

- Keine Redundanz: Jedes Dokument liegt ausschließlich in einem Systemknoten vor
- Temporäre Redundanz: Caching in anderen Systemknoten
- Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung
- Vollständige Redundanz: Jedes Dokument liegt in jedem Systemknoten vor.

Art der verfügbaren Dokumente und Inhalte

Eine Akte ist eine Sammlung von Dokumenten mit Bezug auf eine konkrete Person oder Sache. Die Vielfältigkeit der möglichen Inhalte einer Akte, die durch das Aktensystem, dem die Akte angehört, bestimmt wird, gibt über die Fähigkeit der Akte Auskunft, verschiedene fachliche Gebiete abzudecken. Die Typen der Inhalte und Dokumente lassen sich hierbei beliebig grob oder fein strukturieren. Grundsätzlich kommen Dokumente dabei als Textdokumente, strukturierte Dokumente und Bilddokumente vor.

Textdokumente können dabei sowohl das Ergebnis bzw. den Verlauf einer einzelnen Maßnahme dokumentieren als auch als zusammenfassendes Ergebnisdokument für einen längeren Behandlungszeitraum genutzt werden. Ein Textdokument kann zwar eine innere Struktur besitzen, diese Struktur ist allerdings nicht mit technischen Mitteln umgesetzt und kann folglich nicht zur automatisierten Weiterverarbeitung der Inhalte verwendet werden. Bei strukturierten Dokumenten hingegen ist die Struktur durch konkrete Abbildungsvorschriften in eindeutiger Form, z. B. mittels einer Markup-Sprache wie XML hinterlegt. Strukturierte Dokumente können in den Ausprägungsformen proprietär strukturierte Dokumente oder standardisiert strukturierte Dokumente (siehe 2.5.2) vorkommen. Die Felder der strukturierten Dokumente können numerische Messwerte, textuelle Information oder semantisch eindeutige Werte eines kontrollierten Vokabulars enthalten. Bilddokumente können sowohl stehende als auch bewegte Bilder (Bildsequenzen) beinhalten. Außerdem können in Bilddokumenten Metadaten zu den Bildern enthalten sein, die für die inhaltliche Bewertung der Bilddokumente relevant sind.

Unter Berücksichtigung dieser Erkenntnisse werden für das Kriterium „Art der verfügbaren Dokumente und Inhalte“ die folgenden Ausprägungen identifiziert:

- Textdokument
    - Auf eine einzelne Maßnahme bezogenes Textdokument
    - Zusammenfassendes Textdokument
- Strukturiertes Dokument
    - Standardisiert strukturiertes Dokument
    - Proprietär strukturiertes Dokument

- Bilddokument
    - Einzelbilder
        - Mit Metadaten
        - Ohne Metadaten
    - Bildsequenzen
        - Mit Metadaten
        - Ohne Metadaten.

Inhaltlicher Umfang der Akte

Mit dem Kriterium „Inhaltlicher Umfang der Akte“ wird die fachliche Bandbreite bewertet, die durch die technischen Rahmenbedingungen des Aktensystems für die Akte möglich ist. Der inhaltliche Umfang der Akte kann sowohl durch ausschließlich technische als auch durch fachliche, den Behandlungsprozess betreffende Entscheidungen beeinflusst sein. Krankenakten können je nach Abdeckungsgrad als partielle oder umfassende Krankenakten bezeichnet werden (siehe 2.1.1). Die Ausprägung „partielle Krankenakte“ kann durch eine zusätzliche Nennung des Kriteriums, anhand dessen die partielle Akte gebildet wird, weiter spezifiziert werden.

Die aus diesen Überlegungen resultierenden Ausprägungen sind:

- Umfassende Krankenakte
- Institutionsbezogene partielle Krankenakte: enthält bezogen auf eine Institution eine umfassende Krankenakte
- Fallbezogene partielle Krankenakte: enthält bezogen auf einen Behandlungsfall eine umfassende Krankenakte
- Funktionsbezogene partielle Krankenakte: enthält bezogen auf eine Behandlungsleistung eine umfassende Krankenakte (z. B. alle Röntgenbefunde).

Dominierende Muster bzw. Prinzipien der Softwarearchitektur

Ein weiteres Unterscheidungskriterium für verteilte Krankenakten ist die Softwarearchitektur des Aktensystems. Softwarearchitekturen sind grundsätzlich stark von individuellen Unterschieden geprägt. Grundlegende Ideen der Softwarearchitektur werden abstrakt in Form sogenannter Architekturmuster beschrieben. Die individuellen Softwarearchitekturen der einzelnen Aktensysteme können mittels dieser Architekturmuster zu logischen Gruppen zusammengefaßt und somit kategorisiert werden. Eine vollständige Auflistung aller relevanten Architekturmuster als kontrolliertes Vokabular der Ausprägungen des Kriteriums *dominierende Muster der Softwarearchitektur* ist nicht möglich. Aus diesem Grund werden in der folgenden Aufzählung häufig verwendete Vertreter aufgelistet. Die Auflistung kann bei Bedarf beliebig erweitert werden.

- Schichtenarchitektur
- Mikrokernel-Architektur

- Pipes and Filters
- Model-View-Controller
- Komponentenbasierte Architektur
- Serviceorientierte Architektur usw.

Kommunikationsform

Das Unterscheidungskriterium *Kommunikationsform* bezieht sich auf die Art der Initiierung und Durchführung des Datenaustauschs zwischen den Systemknoten. Es hat einen direkten Einfluss auf die Verteilung und Redundanz der medizinischen Dokumente auf den Knoten des Systems.

Einerseits wird, wie in Kapitel 2.1.8 beschrieben, zwischen den Kommunikationsformen synchron und asynchron unterschieden. Andererseits kann auch eine Unterscheidung nach dem die Datenübertragung initiierenden Akteur vorgenommen werden. Wird die Datenübertragung vom späteren Empfänger der Nutzinformation angestoßen, so wird vom Pull-Prinzip gesprochen. Große Verbreitung und Bekanntheit hat diese Methode z. B. in Form des HTTP-Protokolls zum Abruf von Seiten aus dem Internet erlangt; sie befindet sich aber erheblich länger in vielen anderen Request-Reply-Protokollen im Einsatz. Aufgrund der empfängerseitigen Initiierung des Übertragungsvorgangs ist das Pull-Prinzip für die Verwendung mit synchronen Kommunikationsprotokollen optimal. Diese Verbindung ist allerdings nicht zwangsläufig.

Wird die Datenübertragung vom Sender der eigentlichen Nutzinformation initiiert, so spricht man vom Push-Prinzip. Es kann sowohl mittels synchroner als auch mittels asynchroner Kommunikationsprotokolle umgesetzt werden. Die Verwendung asynchroner Mechanismen ist aber deutlich besser geeignet. Werden per Push Patientendaten von einem Systemknoten zu einem oder mehreren anderen übertragen, so ist bereits zum Übertragungszeitpunkt Wissen über den späteren Bedarf in den Zielknoten notwendig. Grundsätzlich führt die Übertragung von Patientendaten via Push zu verstärkt verteilt und gleichzeitig redundant gespeicherten Daten auf den verschiedenen Knoten innerhalb des Systems. Grundsätzlich gibt es für die Betrachtung des Kriteriums *Kommunikationsform* die folgenden Ausprägungen:

- Pull, synchron
- Push, asynchron, ein Sender, ein Empfänger
- Push, synchron, ein Sender, ein Empfänger
- Push, asynchron, ein Sender, mehrere klar adressierte Empfänger (Multicast)
- Push, asynchron, ein Sender, alle erreichbaren Empfänger (Broadcast).

Rollen der Kommunikanten

Bei der Kommunikation zwischen Knoten treten die einzelnen Knoten in verschiedenen Rollen auf. Sind die Rollen der Knoten durch die Systemarchitektur bereits klar als Client- und Serverknoten vorgegeben, so spricht man von Client/Server-Systemen. Können die Knoten bei den einzelnen Kommunikationsfällen abwechselnd sowohl als Client als auch als Server auftreten, so kann man von gleichberechtigten Knoten (Peers) sprechen. Solche

Systeme werden als Peer-to-Peer-Systeme (P2P) bezeichnet. Daraus ergeben sich die folgenden Ausprägungen für die Systemklassifizierung anhand der Rollen der Kommunikanten:

- P2P-System
- Client/Server-System.

Art der Erweiterbarkeit und Veränderbarkeit

Softwaresysteme können grundsätzlich auf unterschiedliche Arten erweiterbar sein. Dabei beeinflusst die Flexibilität der Erweiterungsmechanismen direkt die Anpassbarkeit von Softwaresystemen an lokale oder sich verändernde Gegebenheiten. Grundsätzlich ist jedes Softwaresystem durch die Veränderung seines Codes an neue Gegebenheiten anpassbar. Die Veränderung an einem Softwaresystem durch die Modifikation des Anwendungscodes durchzuführen, bietet die meiste Flexibilität, da so das System selbst grundlegend verändert werden kann. Andererseits führt diese Art der Änderung zumeist zu hohen zeitlichen Aufwänden. Zudem kann diese Form der Erweiterung von Dritten, die keinen Zugriff auf die Codebasis des zu verändernden Softwaresystems haben, nicht genutzt werden. Alternativ dazu können Softwaresysteme, sofern dies bei der Entwicklung eines Systems bereits vorgesehen wurde, per Konfiguration erweitert werden. Konfiguration bedeutet in diesem Zusammenhang, dass Eigenschaften des Systems durch veränderbar gespeicherte Parameter beeinflussbar sind. Die Veränderbarkeit des Systems ist hierbei auf einen während der Systementwicklung vorgesehenen Umfang von Einstellungen beschränkt. Eine weitere Möglichkeit Systeme erweiterbar zu gestalten ist es, das Hinzufügen zusätzlicher Komponenten, sogenannter Plug-ins, zu ermöglichen. Plug-ins sind Software-Komponenten, die durch Konfiguration zu bestehenden Softwaresystemen hinzugefügt werden können, sofern das Softwaresystem und die Komponente zueinander kompatibel sind. Grundsätzlich können Konfiguration und Plug-ins sowohl als standardisierte Mechanismen als auch als proprietäre (herstellereigene) Mechanismen auftreten. Aus diesem Grund ergeben sich die folgenden Ausprägungen für das Kriterium „Art der Erweiterbarkeit und Veränderbarkeit“:

- Nur durch Veränderung des Programmcodes veränderbar und erweiterbar
- Durch proprietäre Konfiguration veränderbar
- Durch proprietäre Konfiguration veränderbar und erweiterbar
- Durch standardisierte Konfiguration veränderbar und erweiterbar
- Durch Plug-in-Konzept veränderbar und erweiterbar.

Kontext der Erweiterbarkeit und Veränderbarkeit

Neben der Frage, durch welche Mechanismen ein Softwaresystem veränderbar ist, ist auch die Frage, was durch diese Mechanismen verändert werden kann relevant. Bei der Kategorisierung von Softwaresystemen für verteilte Krankenakten wird dazu das Kriterium „Kontext der Erweiterbarkeit und Veränderbarkeit“ verwendet. Prinzipiell kann dieser Kontext beliebig fein auf spezifische Bestandteile konkreter Systeme angewendet werden. Um systemübergreifend eindeutig zu sein, werden für dieses Kriterium zwei für beliebige Systeme verwendbare Ausprägungen definiert.

Ausprägungen:

- Erweiterbarkeit der Funktionalität
- Erweiterbarkeit der Datenstrukturen.

Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts

Ein weiteres Kriterium zur Unterscheidung einzelner Systeme und Projekte aus dem Themengebiet verteilter Krankenakten ist die Unterscheidung nach der Tiefe bzw. dem Detaillierungsgrad ihres Autorisierungskonzepts. Hier erfolgt die Unterscheidung anhand des feingranularsten der Objekte, für das eine eigene Berechtigungsprüfung durchgeführt wird. Für den Vergleich von Systemen in Kapitel 2.6.3 sind deshalb die folgenden Ausprägungen definiert:

- Patientenbasierte Autorisierung
- Fallbasierte Autorisierung
- Transaktionsbasierte Autorisierung
- Dokumentenbasierte Autorisierung
- Dokumentenbestandteilbasierte Autorisierung.

Art der Integration in die bestehende System- und Prozesslandschaft

Die Art und Tiefe der Integration eines Systems zur verteilten Nutzung von Krankenakten in die bestehende System- und Prozesslandschaft eines Gesundheitsdienstleisters beeinflusst direkt die Aufwände, die zur Nutzung verteilter Krankenakten notwendig sind. Aus diesem Grund ist die Art der Integration in die bestehende System- und Prozesslandschaft ein relevantes Kriterium zur Unterscheidung einzelner Systemkonzepte. Für die Bewertung der Integration werden die zwei unterschiedlichen Hauptaktivitäten bei der Nutzung verteilter Krankenaktensysteme, das Zur-Verfügung-Stellen und das Konsumieren von Akteninhalten, getrennt voneinander betrachtet.

Nutzerseite:
- Keine Integration: manuelle Übertragung oder Medienbruch
- Teilweise integriert: automatisierte Integration der Daten
- Vollständig integriert: automatisierte und semantische Integration der Daten

Anbieterseite:
- Keine Integration: manuelle Übertragung
- Teilweise integriert: automatisierte Integration der Daten
- Vollständig integriert: automatisierte und semantische Integration der Daten.

Umfasst die Integration neben einer vollständigen Integration von Nutzer- und Anbieterseite weitere die Prozesse betreffende Merkmale, so wird zusätzlich die folgende Ausprägung verwendet:

- Vollständige Prozessintegration

Alternativ zu den hier vorgestellten Klassifizierungskriterien bieten unter anderen Bergmann u. a. (Bergmann u. a. 2007:S. 131, Table 1) ein alternatives Klassifizierungskonzept für elektronische Kranken- und Patientenakten an, das etwas weniger detailliert ist. Es überdeckt sich in Bezug auf einzelne Kriterien mit dem hier vorgestellten Konzept und teilt die Kriterien in einen Enterprise Viewpoint, einen Information Viewpoint und einen Computational Viewpoint auf. Für die in Kapitel 2.6 durchgeführte Untersuchung ist es nicht geeignet, da es relevante Aspekte wie z. B. die Gestaltung der Softwarearchitektur nicht berücksichtigt und den Aspekt der Integration hinsichtlich der Nutzer und Anbieterseite nicht getrennt behandelt.

### 2.6.2. Durchführung der Untersuchung

Das Konzept zur Durchführung der Untersuchung über die Vielfalt der Ausprägungen und Eigenschaften von verteilten Krankenaktensystemen ist dreistufig aufgebaut. Die erste Stufe umfasst die Auswahl geeigneter Suchbegriffe. Die zweite Stufe beinhaltet die Auswahl einer geeigneten Quellmenge zu untersuchender Projekte, einschließlich ihrer zugänglichen Dokumentation. Die dritte Stufe der Untersuchung besteht aus der Bewertung der in dieser Quellmenge enthaltenen Systeme. Sie wird anhand der Kriterien aus Kapitel 2.6.1 durchgeführt und beinhaltet eine quantitativen Auswertung der einzeln erfassten Werte.

Auswahl geeigneter Suchbegriffe

Im Rahmen der vorgestellten Untersuchung sollen Systeme, die den Kriterien der Begriffsdefinition einer verteilten Krankenakte entsprechen, betrachtet werden. Deshalb basiert die Auswahl der Suchbegriffe auf der Definition des Begriffs sowie verwandten englischsprachigen Synonymen. Aus der Begriffsdefinition in Kapitel 2.2.2 lassen sich neben der verteilten Krankenakte die Begriffe „elektronische Krankenakte“ und „elektronische Patientenakte“ sowie ihre englischen Synonyme „electronic patient record“ und „electronic health record“ als geeignete Suchbegriffe identifizieren. Laut der Definition von Haas (Haas 2006:S.1 des Vorworts) umfasst auch der Begriff „Gesundheitstelematik“ ein inhaltlich den verteilten Krankenakten verwandtes Themengebiet.

Verwendete Suchbegriffe:

- Verteilte Krankenakte
- Distributed Health Record
- Elektronische Krankenakte
- Elektronische Patientenakte
- Elektronische Fallakte
- Care Record
- Health Record
- Patient Record.

Auswahl der Quellmenge

Die Ermittlung der zu bewertenden Projekte und Konzepte erfolgt auf Basis der ausgewählten Suchbegriffe mittels einer Suchmaschine für wissenschaftliche Publikationen. Als Such-

maschine wird der Dienst Google Scholar[11] verwendet. Die Suchanfragen werden mit den dort standardmäßig herrschenden Voreinstellungen durchgeführt. Für jeden Suchbegriff werden die jeweils ersten 50 Ergebnisse ausgewertet. Aus der Anwendung der vorgegebenen acht Suchbegriffe resultieren folglich 400 auszuwertende Treffer. Jeder der 400 Treffer wird durch Lesen des zugänglichen Inhalts mit der Definition verteilter Krankenakten abgeglichen. Befindet sich das beschriebene System innerhalb der Definition wird es in die Menge der zu untersuchenden Systeme aufgenommen.

Die Auswahl von Google Scholar ist dadurch zu begründen, dass sich in vorab durchgeführten Tests zeigte, dass z. B. die wissenschaftliche Datenbank Web of Knowledge häufiger Publikationen zu einzelnen innovativen, in den Softwaresystemen verwendeten Mechanismen, aber seltener zu den Architekturen der Softwaresysteme selbst lieferte. Die Fachdatenbanken von ACM und IEEE, die vorwiegend Publikationen aus der Informatik und Elektrotechnik enthalten, lieferten im gleichen Test mit den Suchbegriffen „Health Record“ und „Elektronische Patientenakte“ auf der ersten Trefferseite erkennbar weniger verwertbare Ergebnisse. Da die in diesem Kapitel vorliegende Untersuchung die Bandbreite verwendeter Lösungsansätze bewerten soll, ist primär die Anzahl der verwertbaren Ergebnisse relevant, zumal die Qualität der Quellen im Rahmen der Analyse der einzelnen Systeme nochmals unabhängig von dem zur Suche verwendeten Werkzeug bewertet wird.

Nach dem Ausschluss der Publikationen, die keine Projekte oder keine vollständigen Konzepte für verteilte Krankenakten beschreiben, werden Treffer zusammengefasst, die bereits ermittelte Systeme behandeln. Die Suche wurde am 25.02.2011 durchgeführt. Die entstandene Ergebnismenge umfasst 30 unterschiedliche Systeme oder Konzepte. Von den ermittelten 30 Systemen mussten vier weitere Systeme unberücksichtigt bleiben, da die zugehörigen Publikationen über die an der Universität Regensburg verfügbaren Abonements elektronischer Zeitschriften nicht zugänglich waren. Folglich stehen für die Untersuchung insgesamt 26 relevante Systeme und Konzepte zur Verfügung.

Durchführung der Untersuchung

Die Erfassung der Untersuchungsergebnisse erfolgt mittels einer eigens dafür entwickelten Datenbankanwendung. Um die Bewertung nach den gegebenen Kriterien zuverlässiger zu gestalten, wird vor der Durchführung der Bewertung bei jedem System oder Konzept nach zusätzlichen Publikationen gesucht, die eventuell weitere Detailinformationen in die Bewertung einbringen könnten. Danach wird das Projekt anhand der in Kapitel 2.6.1 definierten zwölf Kriterien mit ihren gegebenen Ausprägungen klassifiziert. Sollte die Zuordnung zu einer gegebenen Merkmalsausprägung nicht möglich sein, so sieht die Bewertungsdatenbank bei jedem Merkmal die Ausprägungen „sonstige“ und „unbekannt“ vor. Eine „sonstige“ Ausprägung ist dabei eine im Rahmen der zugehörigen Publikationen beschriebene, aber bei den Ausprägungen der Kriterien nicht berücksichtigte Form. Der Wert „unbekannt“ sagt hingegen aus, dass die Ausprägung des entsprechenden Kriteriums für das untersuchte System nicht ermittelt werden konnte. Eine besondere Häufung des Wertes „sonstige“ ist ein Hinweis auf eine ungeeignete Vorauswahl der möglichen

[11] http://scholar.google.de

Merkmalsausprägungen. Ein oftmaliges Auftreten der Ausprägung „unbekannt“ weist auf eine geringe Berücksichtigung des betreffenden Merkmals bei der Konzeption oder Beschreibung der ausgewählten verteilten Krankenaktensysteme hin. Da die Untersuchung auf Veröffentlichungen zu den jeweiligen Systemen basiert, sind teilweise auch aus diesem Grund nicht bei jedem einzelnen System zu allen definierten Kriterien Werte verfügbar.

### *2.6.3. Ergebnisse*

Im Rahmen der Untersuchung wurden 26 Systeme oder Konzepte untersucht. Die Ergebnisse der Untersuchung sind, gegliedert nach den Klassifizierungskriterien aus 2.6.1 dargestellt und erläutert. Das Ziel der Untersuchung ist, die technische und inhaltliche Vielfalt von Projekten und Konzepten für verteilte Krankenaktensysteme zu ermitteln. Informationen über die bewerteten Systeme, die verwendeten Quellen und die einzelnen Auswertungsergebnisse befinden sich im Anhang B dieser Arbeit.

Verbreitungsgebiet der Krankenakte

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Regionaler Verbund: regionaler abgegrenzter Verbund von Einrichtungen | 7 | 26,92% |
| Nicht im Einsatz (Konzept) | 6 | 23,08% |
| Verbund: rechtlich abgegrenzter Verbund von Einrichtungen | 5 | 19,23% |
| Testbetrieb | 2 | 7,69% |
| Unbekannt | 2 | 7,69% |
| Lokal: innerhalb einer Einrichtung | 1 | 3,85% |
| International (einzeln): verschiedene Einrichtungen weltweit | 1 | 3,85% |
| National: alle Einrichtungen eines Landes | 1 | 3,85% |
| Regional: alle Einrichtungen einer Region | 1 | 3,85% |
| **Summe** | **26** | **100,00%** |

Tabelle 4: Verbreitungsgebiet der Krankenakte

Das erste untersuchte Kriterium befasst sich mit dem Verbreitungsgebiet der einzelnen Aktenprojekte oder Aktenkonzepte. Tabelle 4 zeigt, dass ein Großteil der betrachteten Systeme sich entweder in regional oder rechtlich abgegrenzten Verbünden, im Testbetrieb oder noch in der Konzeptphase befindet. Das bedeutet, dass eine praktische Nutzung der betreffenden Systeme nur in beschränktem Umfang stattfindet. Aus diesem Grund sind die in den Tabellen 3 bis 15 dargestellten Daten für die Ermittlung von Zusammenhängen zwischen den Kriterien, die einen Bezug zu den Eigenschaften des praktischen Betriebs aufweisen, und der Nutzung von verteilten Krankenakten nicht verwendbar.

Als Folgerung aus den Zahlen der Tabelle 4 kann die Erkenntnis gezogen werden, dass im Bereich der geografischen Größe und Ausdehnung der Systeme – trotz der Häufung im Bereich kleinerer abgegrenzter Systeme – eine hohe Diversität vorherrscht, die bei der Gestaltung eines allgemeingültigen Konzepts für die Entwicklung verteilter Krankenakten berücksichtigt werden muss.

Menge der Knoten mit eigener Datenspeicherung

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Vollständig dezentrale Datenspeicherung: Jeder Knoten speichert seine Daten selbst | 9 | 34,62% |
| Dezentrale Server: Mehrere spezialisierte Knoten speichern die Daten | 8 | 30,77% |
| Zentrale Datenspeicherung: ein zentraler Knoten, der die Daten speichert | 5 | 19,23% |
| Unbekannt | 4 | 15,38% |
| **Summe** | **26** | **100,00%** |

Tabelle 5: Menge der Knoten mit eigener Datenspeicherung

Das Kriterium „Menge der Knoten mit eigener Datenspeicherung“ dient der Ermittlung der Häufigkeit der Verwendung von zentraler- bzw. dezentraler Datenspeicherung bei der Umsetzung verteilter Krankenakten. Dabei zeigt sich, dass die drei möglichen Ausprägungen in Tabelle 5 etwa zu gleichen Teilen Verwendung finden. Folglich sind alle drei Ansätze gleichermaßen relevant.

Redundanz

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Unbekannt | 12 | 46,15% |
| Keine Redundanz: Jedes Dokument liegt ausschließlich in einem Systemknoten vor | 6 | 23,08% |
| Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung | 5 | 19,23% |
| Vollständige Redundanz: Jedes Dokument liegt in jedem Systemknoten vor | 3 | 11,54% |
| **Summe** | **26** | **100,00%** |

Tabelle 6: Redundanz der Daten in verteilten Krankenakten

Die Auswertung des Kriteriums „Redundanz“ zeigt, dass in den Publikationen zu den untersuchten Systemen konkrete Aussagen zur redundanten oder nichtredundanten Speicherung von Dokumenten häufig fehlen. Auch Indizien, die eine Eingruppierung dennoch ermöglichen sind bei 12 der 26 Systeme nicht ermittelbar. Die Betrachtung der verbleibenden 14 Systeme weißt auf eine Tendenz zu keiner bzw. geringer Redundanz der Dokumente hin. Vollständige Redundanz wird vor allem in Systemen erreicht, die Daten ausschließlich aktiv an potenzielle Empfänger verteilen. Dazu ist es nötig, Dokumente vorab an alle Zielknoten zu verteilen, in denen das Dokument später eventuell genutzt werden soll.

Aus der Betrachtung der Tabelle 6 resultiert, dass eine Entwicklungsmethode für verteilte Krankenakten neben Konzepten für den effizienten entfernten Zugriff auf nichtredundante Dokumente auch zwingend Konzepte für die Erhaltung der Konsistenz aller redundant verteilten Replikate und der zugehörigen originalen Dokumenten enthalten muss.

Art der verfügbaren Dokumente und Inhalte

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Beliebige Dokumente | 15 | 53,57% |
| Strukturiertes Dokument: standardisiert | 10 | 35,71% |
| Bilddokument: Bildsequenz: mit Metadaten | 2 | 7,14% |
| Strukturiertes Dokument: proprietär | 1 | 3,57% |
| **Summe** | **28** | **100,00%** |

Tabelle 7: Art der verfügbaren Dokumente und Inhalte

Zwei der untersuchten Systeme sind bei dem Kriterium „Art der verfügbaren Dokumente und Inhalte“ mit je zwei Ausprägungen klassifiziert. Bei einer Klassifizierung mit der Ausprägung „beliebige Dokumente“ sind grundsätzlich keine weiteren Ausprägungen vergeben worden.

Die Mehrheit der untersuchten Systeme ist fähig, beliebige Dokumente zu verarbeiten. Das bedeutet, dass die betreffenden Systeme prinzipiell für alle in Kapitel 2.6.1 definierten Arten von Dokumenten einsetzbar sind. Ungeklärt ist dabei, ob es sich bei der Eignung um eine Verarbeitung der Dokumenteninhalte oder ein Zur-Verfügung-Stellen der Dokumente auf Basis von Metadaten handelt. Des Weiteren ist ein großer Anteil der Systeme auf die Verarbeitung standardisierter strukturierter Dokumente oder Akten der Standards HL7 CDA oder CCR spezialisiert. Eine weitere Gruppe sind Bilddokumente mit Metadaten. Dabei handelt es sich um DICOM-Dateien, deren Metadaten-Bestandteil ebenfalls standardisiert strukturiert ist. Durch die häufige explizite Nennung von DICOM-Dateien in den Publikationen zu Systemen, die hier mit der Ausprägung „beliebige Dokumente“ klassifiziert sind, besitzt auch diese Ausprägung eine hohe Relevanz.

Die Entwicklung eines Systems, das genau einen Standard unterstützt, ist daher durch die Entwicklungsmethode ebenso zu berücksichtigen wie die Entwicklung flexibler Systeme, die gleichzeitig mit Dokumenten verschiedener Standards verwendet werden können.

Inhaltlicher Umfang der Akte

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Umfassende Krankenakte | 8 | 30,77% |
| Funktionsbezogene partielle Krankenakte | 7 | 26,92% |
| Unbekannt | 6 | 23,08% |
| Institutionsbezogene partielle Krankenakte | 3 | 11,54% |
| Fallbezogene partielle Krankenakte | 2 | 7,69% |
| **Summe** | **26** | **100,00%** |

Tabelle 8: Inhaltlicher Umfang der Akte

Bezogen auf den inhaltlichen Umfang der Krankenakten ergibt die Untersuchung ein sehr vielfältiges Bild. Unter den untersuchten Systemen befinden sich sowohl umfassende als auch partielle Akten. Bei den partiellen Akten sind die drei Formen institutionsbezogen, fallbezogen und funktionsbezogen partiell im Untersuchungsergebnis vertreten.

Die acht identifizierten umfassenden Krankenakten müssen mit der Einschränkung betrachtet werden, dass sie sich in ihrer Mehrzahl noch nicht im vollständig produktiven Einsatz befinden oder nur in abgegrenzten regionalen Verbünden eingesetzt werden (siehe Tabelle 9).

| Umfang | Verbreitung | Anzahl |
|---|---|---|
| Umfassende Krankenakte | Nicht im Einsatz (Konzept) | 3 |
| Umfassende Krankenakte | Regionaler Verbund: regionaler abgegrenzter Verbund von Einrichtungen | 2 |
| Umfassende Krankenakte | Regional: alle Einrichtungen einer Region[12] | 1 |
| Umfassende Krankenakte | National: alle Einrichtungen eines Landes[13] | 1 |
| Umfassende Krankenakte | International (einzeln): verschiedene Einrichtungen weltweit | 1 |

Tabelle 9: Zusammenhang - Umfang und Verbreitung der Systeme bei umfassenden Akten

Fallbezogene partielle Krankenakten und funktionsbezogene partielle Krankenakten stellen andere Anforderungen an ein System als umfassende und institutionsbezogene partielle Krankenakten. Fallbezogene Krankenakten benötigen die Zuordnung von relevanten Dokumenten zu Behandlungsfällen als Grundlage für die Vergabe von Zugriffsrechten und die Bereitstellung von Dokumenten. Funktionsbezogene Krankenakten können häufig auf eine relativ kleine Zahl von Standards und Eigenschaften reduziert werden. Die daraus resultierenden Unterschiede müssen bei einer für verteilte Krankenakten allgemeingültigen Entwicklungsmethode berücksichtigt werden.

Dominierende Muster bzw. Prinzipien der Softwarearchitektur

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Unbekannt | 10 | 35,71% |
| Message Broker | 4 | 14,29% |
| Schichtenarchitektur | 4 | 14,29% |
| Komponentenbasierte Architektur | 3 | 10,71% |
| Serviceorientierte Architektur | 3 | 10,71% |
| Mikrokernel-Architektur | 2 | 7,14% |
| Software-Agenten | 1 | 3,57% |
| Sonstige | 1 | 3,57% |
| **Summe** | **28** | **100,00%** |

Tabelle 10: dominierende Muster bzw. Paradigmen der Softwarearchitektur

Bei 10 von 26 untersuchten Systemen wurden keine Angaben zur verwendeten Softwarearchitektur gemacht. Bei den verbleibenden 16 Systemen ist eine große Bandbreite an möglichen Ausprägungen erkennbar. Die Softwarearchitekturen von zwei Systemen sind durch eine Kombination von je zwei der aufgelisteten Muster und Prinzipien geprägt.

[12] Das System befindet sich noch im Aufbau; der angegebene Wert entspricht der geplanten Endausbaustufe.
[13] Das System befindet sich noch in der Konzeptionsphase; der angegebene Wert entspricht der geplanten Endausbaustufe.

Eine allgemeingültige Entwicklungsmethode für verteilte Krankenakten sollte folglich den gesamten Umfang der für verteilte Systeme relevanten Softwarearchitekturprinzipien berücksichtigen. Message Broker und Schichtenarchitektur bzw. serviceorientierte Architektur, als die drei am stärksten vertretenen Architekturansätze, legen nahe, die asynchrone nachrichtenbasierte und die synchrone Request-Reply-basierte Kommunikation bei der Gestaltung der Entwicklungsmethodik gesondert zu behandeln.

Kommunikationsform

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Pull, synchron | 13 | 43,33% |
| Unbekannt | 7 | 23,33% |
| Push, asynchron, ein Sender, ein Empfänger | 7 | 23,33% |
| Push, synchron, ein Sender, ein Empfänger | 2 | 6,67% |
| Push, asynchron, ein Sender, mehrere klar adressierte Empfänger | 1 | 3,33% |
| **Summe** | **30** | **100,00%** |

Tabelle 11: Kommunikationsform(en)

Die Implementierung eines Systems kann mehrere Ausprägungen des Kriteriums *Kommunikationsform* beinhalten, da es unterschiedliche Kommunikationsbeziehungen besitzen kann. Aus diesem Grund ist den Systemen teilweise nicht nur eine Ausprägung zugeordnet. Sowohl das Push- als auch das Pull-Prinzip sind in der untersuchten Menge von Systemen in relevanter Anzahl vorhanden. Die Erkenntnis, die aus diesem Ergebnis gezogen werden kann, deckt sich zum Teil mit dem Ergebnis der Untersuchung zur Softwarearchitektur (siehe Tabelle 10). Eine jeweils eigenständige und gleichberechtigte Betrachtung von synchronen und asynchronen Kommunikationsmechanismen sowie unterstützenden Verfahrensweisen ist für eine allgemeingültige Entwicklungsmethode notwendig.

Rollen der Kommunikanten

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Client / Server | 16 | 61,54% |
| Unbekannt | 7 | 26,92% |
| Peer (in P2P System) | 3 | 11,54% |
| **Summe** | **26** | **100,00%** |

Tabelle 12: Rollen der Kommunikanten

Bei 16 der 26 untersuchten Systeme besteht zwischen den beteiligten Kommunikanten eine eindeutige Verteilung der Rollen Client und Server. Die Publikationen zu sieben der untersuchten Systeme liefern keine eindeutigen Hinweise, während in drei Systemen die Kommunikanten als gleichberechtigte Peers betrachtet werden. In einem Peer-to-Peer-System wechseln die Rollen Client und Server dynamisch mit jeder einzelnen Kommunikationsbeziehung. Das bedeutet, jeder Peer kann, nachdem er bei einem Aufruf als Client fungiert hat, im nächsten Aufruf als Server fungieren. Folglich existieren auch in einem Peer-to-Peer-System temporäre Client-Server-Beziehungen. Hinsichtlich der Kommunikation

zwischen zwei Knoten ist aus diesem Grund keine besondere Unterscheidung zwischen Client-Server und Peer-to-Peer nötig.

Bei der Ermittlung des für eine konkrete Anfrage geeigneten Kommunikationspartners unterscheiden sich die Mechanismen klassischer Client-Server-Systeme von denen, die in Peer-to-Peer-Systemen zum Einsatz kommen. Entsprechende Mechanismen müssen in einer allgemeingültigen Entwicklungsmethode für verteilte Krankenaktensysteme berücksichtigt werden.

Art der Erweiterbarkeit und Veränderbarkeit

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Unbekannt | 15 | 53,57% |
| Standardisierte Konfiguration, veränderbar und erweiterbar | 7 | 25,00% |
| Programmcode | 3 | 10,71% |
| Proprietäre Konfiguration, veränderbar und erweiterbar | 2 | 7,14% |
| Plug-in-Konzept | 1 | 3,57% |
| **Summe** | **28** | **100,00%** |

Tabelle 13: Art der Erweiterbarkeit und Veränderbarkeit

Eine flexible Anpassung von IT-Systemen an sich verändernde Gegebenheiten kann durch verschiedene Mechanismen realisiert werden. In den Publikationen zu 15 der 26 untersuchten Systeme werden keine Aussagen über die Erweiterbarkeit und Veränderbarkeit getroffen. Die verbleibenden elf Systeme sind durch unterschiedliche Mechanismen veränderbar gestaltet.

Eine Anpassung durch die Modifikation des Quellcodes ist grundsätzlich für jedes beliebige Softwaresystem möglich. Sie kann in der Praxis jedoch nur dann durchgeführt werden, wenn der Quellcode der an einer Änderung interessierten Benutzergruppe zur Verfügung steht. Aus diesem Grund sind in Tabelle 13 nicht alle Systeme dieser Merkmalsausprägung zugeordnet.

Der größte Teil der bezüglich Erweiterbarkeit und Veränderbarkeit erfolgreich untersuchten Systeme ist durch standardisierte Konfigurationsmechanismen veränderbar und erweiterbar. Diese Merkmalsausprägung subsumiert die Abbildung und Modifikation konkreter Dokumente innerhalb abstrakter standardisierter Modelle, wie z. B. die Modellierung von HL7-CDA-Dokumenten oder die Spezifikation von Archetypen in ISO 13606. Ist ein System geeignet, beliebige, innerhalb eines definierten Standards modellierte Dokumente zu verarbeiten, so wird es in Tabelle 13 als standardisiert konfigurierbar bezeichnet.

Neben standardisiert konfigurierbaren und via Quellcode veränderbaren Systemen sind in der untersuchten Quellmenge auch proprietär konfigurierbare und per Plug-in-Konzept erweiterbare Systeme enthalten. Als proprietär konfigurierbar werden jene Systeme bezeichnet, die durch nicht standardisierte systemeigene Konfigurationsparameter innerhalb eines vom Hersteller vorgesehenen Rahmens verändert werden können. Als Plug-in-Konzept wird ein Konzept bezeichnet, das es ermöglicht zusätzlichen ausführbaren Code in Form von Softwarekomponenten per Konfiguration zu einem bestehenden Softwaresystem hinzuzufügen.

Konfigurations- und Plug-in-Mechanismen sind allgemein bekannte Lösungsansätze zur Schaffung flexibel anpassbarer Softwaresysteme. In verteilten Krankenakten ist darüber hinaus wichtig bei der Umsetzung von Mechanismen zur Verbesserung der Flexibilität einen besonderen Schwerpunkt auf die Integration existierender Standards zu legen.

Kontext der Erweiterbarkeit und Veränderbarkeit

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Unbekannt | 15 | 48,39% |
| Datenstrukturen | 10 | 32,26% |
| Funktionalität | 6 | 19,35% |
| **Summe** | **31** | **100,00%** |

Tabelle 14: Kontext der Erweiterbarkeit und Veränderbarkeit

Während das Kriterium „Art der Erweiterbarkeit“ beschreibt, wie ein System erweiterbar gestaltet ist, wird durch das Kriterium „Kontext der Erweiterbarkeit“ beschrieben, was an dem jeweiligen System erweiterbar gestaltet ist. Von den elf Systemen, zu denen in Publikationen Aussagen bezüglich der Erweiterbarkeit und Veränderbarkeit getroffen werden, sind zehn hinsichtlich ihrer Datenstrukturen veränderbar. Fünf davon lassen sich auch in ihrer Funktionalität verändern. Eines der untersuchten Systeme ist ausschließlich bezüglich seiner Funktionalität, nicht aber bezüglich seiner Datenstrukturen erweiterbar gestaltet.

Unter einer Erweiterung oder Veränderung der Funktionalität wird in diesem Zusammenhang eine Veränderung am Verhalten des Systems verstanden. Das Hinzufügen einer neuen Kommunikationsbeziehung, eines zusätzlichen Kommunikationsprotokolls oder auch das Hinzufügen zusätzlicher Masken sind Beispiele für veränderte Funktionalität.

Die Untersuchung zeigt, dass die Veränderbarkeit der Datenstrukturen und der Funktionalität gleichermaßen relevant sind, obwohl in den meisten Standards ausschließlich die Veränderbarkeit von Datenstrukturen berücksichtigt wird. Folglich muss eine für beliebige verteilte Krankenaktensysteme verwendbare Entwicklungsmethode beide Aspekte berücksichtigen, um allgemeingültig sein zu können.

Detaillierungsgrad des Autorisierungskonzepts

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Unbekannt | 17 | 65,38% |
| Dokumentenbasiert | 5 | 19,23% |
| Dokumentenbestandteilbasiert | 2 | 7,69% |
| Fallbasiert | 1 | 3,85% |
| Patientenbasiert | 1 | 3,85% |
| **Summe** | **26** | **100,00%** |

Tabelle 15: Detaillierungsgrad des Autorisierungskonzepts

Obwohl verschiedene Autoren die Vertraulichkeit von medizinischen Dokumenten als wichtige Anforderung an elektronische Kranken- und Patientenakten betrachten (vgl. z. B. Brunner & Kaiser 2008:S. 72), werden in den Publikationen nur bei 9 der 26 untersuchten Systeme Angaben zu einem darin enthaltenen Autorisierungskonzept gemacht. Diese Tatsache ist auf das Problem zurückzuführen, dass in Systemen, die Dokumente aktiv verteilen (Push-Prinzip), systemseitig kein klassisches Autorisierungskonzept umgesetzt werden kann. Die Autorisierung für den Zugriff auf ein Dokument erfolgt vielmehr vor dem Versenden durch die Auswahl von Personen oder Einrichtungen als Empfänger eines Dokuments.

Bei den neun Systemen mit klassifizierbarem Autorisierungskonzept zeigt sich bei den feingranularen Konzepten eine Häufung. Zugriffsberechtigungen, die auf der Ebene von Dokumenten oder Dokumentenbestandteilen vergeben werden, ermöglichen maximale Flexibilität bei der Erreichung datenschutzkonformer Zugriffsstrukturen. Gleichzeitig führen Sie allerdings zu größeren systemseitigen Aufwänden als die grobgranulareren Ansätze auf Fall- oder Patientenebene.

Die Untersuchung zeigt, dass unterschiedlich fein abgestimmte Autorisierungskonzepte für die Entwicklung verteilter Krankenakten relevant sind. Auch Konzepte zur Restriktion des Zugriffs auf bereits versendete Dokumente können, sofern verfügbar, ein mögliches Gesamtkonzept sinnvoll erweitern.

Art der Integration in die bestehende System- und Prozesslandschaft

| Ausprägung | Anzahl | Prozent |
|---|---|---|
| Vollständige Prozessintegration | 3 | 8,11% |
| Unbekannt | 8 | 21,62% |
| Nutzerseitig: teilweise Integration | 8 | 21,62% |
| Nutzerseitig: keine Integration | 3 | 8,11% |
| Anbieterseitig: vollständige Integration | 7 | 18,92% |
| Anbieterseitig: teilweise Integration | 6 | 16,22% |
| Anbieterseitig: keine Integration | 2 | 5,41% |
| **Summe** | **37** | **100,00%** |

Tabelle 16: Art der Integration in die bestehende System- und Prozesslandschaft

Die Art und Weise, in der ein System in eine bestehende System- und Prozesslandschaft integriert werden soll, beeinflusst maßgeblich die dafür verwendbaren Mechanismen. Je nahtloser ein System integriert werden soll, desto größer ist der daraus resultierende Einfluss auf verschiedene Eigenschaften des Systems selbst. Zur Erreichung einer vollständigen, also technischen, syntaktischen und semantischen Integration ist zwischen den Systemen ein exaktes gemeinsames Verständnis der Strukturen und Inhalte oder zumindest ein Mechanismus zur automatisierten Abbildung zweier unterschiedlicher Konzepte erforderlich. Bei einer teilweisen Integration werden mindestens die technischen Mechanismen zur Übertragung der Daten benötigt. Diese können auch unabhängig von den übertragenen Inhalten sein.

Die Untersuchung zeigt, dass bei den betrachteten Systemen bezüglich der erreichten Integration große Unterschiede existieren. Tendenziell sind mehr Systeme anbieterseitig als nutzerseitig integriert. Innerhalb der Publikationen zu acht der bewerteten Systeme sind keine Aussagen zur Integration in bestehende Landschaften getroffen.

Grundsätzlich lässt sich daraus ableiten, dass beliebige Formen von Integrationsmechanismen für eine allgemeine Entwicklungsmethode für verteilte Krankenakten relevant sind. Aufgrund der erkennbaren Defizite bei der nutzerseitigen Systemintegration können Verbesserungen bei der Verwendung von Integrationsmechanismen und eine systematische Darstellung derselben zu Verbesserungen bei der Nutzbarkeit verteilter Krankenakten führen.

# 3. Auswahl der Entwicklungsmethodik

Durch den Begriff *Methode* wird eine wissenschaftliche, planmäßige oder folgerichtige Art des Vorgehens beschrieben.[14] Der theoretische Hintergrund bzw. die Lehre bezüglich der Anwendung einer Methode wird als *Methodik* bezeichnet (Oestereich 2009:S. 19). Sie umfasst also die Anwendungsprinzipien einer Methode. Bezogen auf die Entwicklung von Softwaresystemen beschreibt eine Entwicklungsmethodik eine Methode zur planmäßigen Entwicklung von Software.

Ziel dieser Arbeit ist die Entwicklung einer Methode, die es ermöglicht, systematisch flexible Softwaresysteme für verteilte Krankenakten zu entwickeln. Die zugehörige Methodik muss die Erweiterbarkeit und Anpassung der Methode auf der Basis neuer Erkenntnisse aus Forschung und Praxis beinhalten. Gleichzeitig muss sie auch die Verwendbarkeit der entstandenen Systeme für wissenschaftliche Untersuchungen verbessern.

In Kapitel 2 werden die Rahmenbedingungen und Anforderungen für Softwaresysteme zur Realisierung verteilter Krankenakten untersucht. Die dort betrachteten Anwendungsfälle, Anforderungen, Standards und Projekte bilden, neben den in Kapitel 3.1 genannten nicht domänenspezifischen Anforderungen, die Entscheidungsgrundlage für die Auswahl und Gestaltung einer geeigneten Entwicklungsmethodik.

Software Engineering ist eine Teildisziplin der Informatik und subsumiert als Begriff die Menge der Methoden zur systematischen bzw. ingenieurmäßigen Planung und Entwicklung von Softwaresystemen. Im deutschen Sprachgebrauch wird *Software Engineering* häufig auch mit dem Begriff *Softwaretechnik* bezeichnet (GI FG Softwaretechnik 2010). Es umfasst die Menge der derzeit bereits bekannten und somit möglicherweise geeigneten Methoden zur Entwicklung von Softwaresystemen. Die Menge der Methoden des Software Engineering besteht jedoch nicht generell aus einander widersprechenden oder gegensätzlichen Elementen. Stattdessen sind die Methoden häufig miteinander kombinierbar und können in verschiedenen Phasen des Entwicklungsprozesses einander ergänzend verwendet werden.

Die Auswahl einer Entwicklungsmethodik für verteilte Krankenakten schließt somit die ergänzende Verwendung weiterer Methoden des Software Engineering nicht aus. Die ausgewählte Methodik wird allerdings zum Hauptbestandteil des Konzepts.

## *3.1. Anforderungen an die Entwicklungsmethode*

An eine Entwicklungsmethode für verteilte Krankenakten werden vielfältige Anforderungen gestellt. Einerseits soll die Methode entsprechend der in Kapitel 2 beschriebenen Anforderungen und Rahmenbedingungen geeignet sein, die Planung und Entwicklung von verteilten Krankenakten zu vereinfachen. Aus der Vereinfachung soll auch eine Verbesserung der qualitativen, also nichtfunktionalen Eigenschaften der entstandenen Systeme resultieren. Andererseits soll die kontinuierliche Verbesserung der Methode durch sich selbst ermöglicht

[14] Vgl. Duden 1, Die deutsche Rechtschreibung

werden. Erkenntnisse, die zu einer Verbesserung der Entwicklungsmethode führen können, werden durch Forschung erworben. Deshalb muss die Entwicklungsmethode eine Schnittstellenfunktion zwischen Forschung und Entwicklung einnehmen. Die Anforderungen an die der Entwicklungsmethode zugrunde liegende Entwicklungsmethodik werden sowohl durch praktische als auch durch wissenschaftliche Aspekte beeinflusst. Aus diesem Grund werden die Anforderungen an die Entwicklungsmethodik getrennt nach den Einflussgrößen Wissenschaft und Praxis betrachtet.

### 3.1.1. Entwicklung verteilter Krankenakten

Die Entwicklung verteilter Krankenakten benötigt eine Entwicklungsmethode, die für die gesamte Menge der Systeme seiner Kategorie anwendbar ist. Dabei sind insbesondere die in Kapitel 2 beschriebenen Erkenntnisse über die Zieldomäne zu berücksichtigen. Sie beinhalten die fachlich bedingten und die technischen Schwerpunkte bei der Entwicklung verteilter Krankenakten und zeigen so die notwendigen Schwerpunkte für eine geeignete Entwicklungsmethodik auf.

Grundlage der Planung und Entwicklung von Softwaresystemen ist die Abbildung anwendungsrelevanter Artefakte der Zieldomäne in problemadäquaten Modellen (vgl. z. B. Rechenberg 2000:S. 202). Die Methoden zur Erstellung dieser Modelle sind sehr vielfältig. Häufig werden Informationssysteme aus der Prozess- und der Datensicht modelliert. Diese Form der Modellierung bildet primär die funktionalen Anforderungen eines Informationssystems ab. Aussagen über nichtfunktionale Eigenschaften des aus den Modellen resultierenden Softwareprodukts können dadurch noch nicht getroffen werden. Kapitel 2.4.2 zeigt, dass die Erfüllung von nichtfunktionalen Anforderungen aus den Bereichen Sicherheit, Datenschutz, Zuverlässigkeit, Erweiterbarkeit, Anpassbarkeit, Veränderbarkeit sowie von speziellen Eigenschaften der Kommunikation eine Voraussetzung für die Einsatzfähigkeit von verteilten Krankenakten ist.

Die Modellierung inhaltlicher und funktionaler Aspekte in Form von Daten- und Prozessmodellen ist bei der Entwicklung medizinischer Informationssysteme inzwischen durch verschiedene Konzepte weitgehend etabliert oder standardisiert. Beispiele für Modellierungsansätze zur Abbildung komplexer medizinischer Datenstrukturen sind der Modellierungsprozess aus HL7v3 und das Archetypenmodell aus openEHR (siehe Kapitel 2.5). Die beiden Beispiele zeigen, dass derzeit für den Aspekt der Datenmodellierung mehrere gleichwertige Methoden existieren. Eine Betrachtung der in Kapitel 2.5.2 beschriebenen Vielzahl an Nachrichten- und Dokumentenstandards zum Austausch strukturierter medizinischer Daten zeigt, dass es daneben noch zusätzlich konkurrierende Standards für den Austausch von strukturierten medizinischen Daten gibt. Ein System, das durchgängig einen der genannten Standards implementiert, ist im Gesamtumfeld verteilter Krankenakten nur bedingt interoperabel, da einzelne für die Kommunikation relevante dritte Systeme auf anderen Standards basieren können.

Aufgrund dieser Tatsachen ist es erforderlich, dass die Entwicklungsmethodik für verteilte Krankenakten in Kombination mit explizit ausgewählten Standards für die Erstellung jeweils geeigneter Systemmodelle verwendet werden kann. Außerdem wird die Fähigkeit benötigt,

Systemmodelle zu generieren, die von den Spezifika der konkurrierenden Standards unabhängig sind und so ggf. mehrere Standards gleichzeitig unterstützen können.

Die Modellierung von Prozessen für verteilte Krankenakten wird durch die Definition einrichtungsübergreifender Behandlungspfade realisiert. Dabei variiert die Gestaltung der Pfade stark zwischen den einzelnen medizinischen Einrichtungen (siehe Kapitel 2.3). Außerdem sind Behandlungspfade auch im Zeitverlauf einer stetigen Veränderung unterworfen. Diese Veränderung kann beispielsweise durch neue Erkenntnisse bezüglich der Optimierung einzelner Behandlungsabläufe oder durch Veränderungen in den vertraglichen Beziehungen zwischen den beteiligten Gesundheitsdienstleistern bedingt sein. Aus diesem Grund sind Behandlungspfade als regelmäßig veränderliche Bestandteile des Anwendungssystems zu betrachten. Ihre Veränderbarkeit muss ein grundlegender Bestandteil des Systemkonzepts sein, sofern Behandlungspfade im System technisch abgebildet sind.

Datenmodell und Prozessmodell sind folglich im Zeitverlauf variable Bestandteile des Systems verteilte Krankenakte. Deshalb ist die zentrale Herausforderung für eine geeignete Entwicklungsmethodik die Fähigkeit zu besitzen, dieser stetigen Veränderbarkeit durch geeignete Konzepte zu begegnen. Als geeignet kann ein Konzept bezeichnet werden, wenn der Aufwand für eine Veränderung in den vorab als veränderlich identifizierten Bereichen möglichst gering ist.

Neben den Eigenschaften der Zieldomäne verteilte Krankenakten beeinflussen auch allgemeine, durch generelle Eigenschaften von Softwareentwicklung bedingte Faktoren die Auswahl und Gestaltung geeigneter Entwicklungsmethoden. So kann die Effizienz der einzelnen Entwickler durch häufige Störungen oder Unterbrechungen des Gedankenflusses stark reduziert werden. Szóstek und Markopoulos zeigen den Einfluss von Unterbrechungen des Gedankenflusses durch Fragen (Face-to-Face Interruptions) bei geistiger oder kreativer Arbeit im Büro (Szostek & Markopoulos 2006). Nakakoji bezieht sich in seinem Artikel „Supporting Software Development as Collective Creative Knowledge Work“ (Nakakoji 2008:S. 2) auf die Erkenntnisse von Szóstek und Markopoulos und betrachtet die Auswirkung von Unterbrechungen im Kontext der Softwareentwicklung als kreative, auf umfassenden Kenntnissen basierende Tätigkeit (Nakakoji 2008:S. 6). Hierbei betrachtet er primär kommunikationsbedingte Unterbrechungen und Störungen. Die Beschreibung des Sachverhalts legt die Annahme nahe, dass ähnliche Zusammenhänge auch mit beliebigen anderen extern initiierten Unterbrechungen kreativer Denkphasen im Softwareentwicklungsprozess bestehen. Deshalb wird nachfolgend davon ausgegangen, dass durch die Entwicklungsmethode erzwungene Tätigkeitswechsel ebenfalls zu einer Unterbrechung des Gedankenflusses führen. Diese Annahme wird zusätzlich durch die stetig zunehmende Verbreitung agiler Methoden der Softwareentwicklung untermauert, da diese Methoden es dem Entwickler erlauben, den Entwicklungsprozess stärker der aktuellen Situation angepasst zu gestalten.

Aus der Annahme, dass auch durch die Entwicklungsmethode erzwungene Tätigkeitswechsel zu einer Unterbrechung des kreativen Gedankenflusses des Entwicklers führen können, resultieren für die Auswahl der Entwicklungsmethodik weitere Anforderungen. Die ausgewählte Methodik soll dem Entwickler einen weitreichenden Spielraum bei der Auswahl der Zeitpunkte und des Umfangs ihres Einsatzes im Rahmen eines konkreten

Entwicklungsprojekts geben. Die Methodik soll mit den verschiedenen derzeit bekannten Vorgehensmodellen für die Softwareentwicklung kombinierbar sein und weder diese selbst, noch den zugehörigen Entwicklungsprozess behindern.

Außerdem muss die Entwicklungsmethode aus Gründen der Praktikabilität einfach anwendbar sein, sodass nur wenige Hürden für ihren praktischen Einsatz überwunden werden müssen. Durch eine möglichst kurze Einarbeitungszeit für die sichere Beherrschung erhöht sich die Wahrscheinlichkeit für den erfolgreichen praktischen Einsatz einer Entwicklungsmethode.

### 3.1.2. Wissenschaftlicher Prozess

Wenn wissenschaftliche Erkenntnisse die Qualität einer Gattung von Softwareprodukten verbessern sollen, müssen sie Aufschluss über Zusammenhänge zwischen beobachtbaren Eigenschaften der Software und deren zur Planungs- und Entwicklungszeit beeinflussbaren Gestaltungsformen und Kombinationsmöglichkeiten der Bestandteile finden und belegen. Um derartige Zusammenhänge zuverlässig zu belegen, ist die Untersuchung einer großen Zahl gleichartiger oder zumindest ähnlicher Systemen mit empirischen Mitteln erforderlich.

Friedmann und Wyatt (Friedman & Wyatt 2006:S. 3) listen Beweggründe für die Durchführung systematischer Untersuchungen an medizinischen Informationssystemen auf. Die angeführten wissenschaftlichen und pragmatischen Gründe dokumentieren die Notwendigkeit für systematische Untersuchungen von medizinischen Informationssystemen durch die Weiterentwicklung der verfügbaren Methoden mittels einer untersuchungsgestützten Bewertung ihrer Eigenschaften.

Kitchenham, Dyba und Jorgensen erläutern in ihrem Aufsatz zu beweisbasiertem Software Engineering (Evidence-based Software Engineering) (Kitchenham u. a. 2004), warum Software Engineering als Disziplin von einer intensiven Kopplung zwischen der empirischen Untersuchung seiner Methoden- und Methodenbestandteile und deren praktischer Nutzung zur Planung und Umsetzung von Softwaresystemen profitieren kann. Ein flächendeckender, praktischer Einsatz von beweisbasiertem Software Engineering ist derzeit aufgrund einiger durch die Autoren beschriebener Faktoren noch nicht möglich. Beispiele für diese Faktoren sind fehlende gemeinsame Leitlinien und Standards für die Durchführung empirischer Untersuchungen über Methoden des Software Engineerings aber auch das Fehlen geeigneter Publikationsformen und -wege für die Ergebnisse (Kitchenham u. a. 2004:S. 4–5).

Aus diesen Erkenntnissen resultiert eine zusätzliche Anforderung an die Methodik zur Entwicklung verteilter Krankenakten. Sie muss durch wissenschaftliche Erkenntnisse systematisch weiterentwickelt werden können. Grundlage dafür ist die Idee des beweisbasierten Software Engineerings, alle Methoden des Software Engineering empirisch auf ihre Tauglichkeit für bestimmte Anwendungszwecke hin zu untersuchen sowie dessen Zielsetzung, Methoden oder deren Bestandteile durch wissenschaftliche Erkenntnisse weiterzuentwickeln.

Eine solche Weiterentwicklung kann allerdings nur dann erfolgen, wenn die existierenden Systeme effizient durch empirische Studien untersucht werden können. Untersucht werden

müssen dabei die Zusammenhänge zwischen gewünschten oder unerwünschten Systemeigenschaften, den gewählten Systembestandteilen und deren Kombination. Eine textuelle Beschreibung ohne geregeltes Vokabular für die Systembestandteile führt, aufgrund der meist nicht eindeutigen Beschreibung derselben, zu großer Ungenauigkeit beim Vergleich von Systemen. Vorhandene Defizite in der Eindeutigkeit existierender Systembeschreibungen werden z. B. durch das Vorkommen unbekannter Werte in den Untersuchungsergebnissen aus Kapitel 2.6 belegt. Für die Durchführung tragfähiger Untersuchungen ist deshalb eine eindeutige Typisierung der Systembestandteile sowie der Kombinationsbeziehungen zwischen den Systembestandteilen notwendig. Eine Entwicklungsmethode, die eine solche Typisierung im Rahmen der Entwicklung eines Systems unterstützt, fördert somit zugleich die empirische Untersuchbarkeit des Systems als Bestandteil einer zu untersuchende Menge von Systemen.

Aus der Notwendigkeit der empirischen Untersuchung großer Mengen von Systemen zur tragfähigen Ermittlung kausaler Zusammenhänge zwischen Eigenschaften und Bestandteilen der Systeme resultiert die Anforderung, dass die gewählte Methodik die verwendeten softwaretechnischen Bestandteile eines Systems eindeutig benennbar macht.

Zusammengefasst bedeuten die Anforderungen, dass die zu findende Methode durch die Integration wissenschaftlicher Erkenntnisse leicht weiterentwickelbar sein muss. Gleichzeitig muss die Methode auch die Generierung wissenschaftlicher Erkenntnisse fördern. Ein wesentlicher Aspekt zur Verbesserung wissenschaftlicher Untersuchbarkeit ist, dass durch die Anwendung der Methode während der Entwicklung verteilter Krankenakten die spätere systematische Vergleichbarkeit der entstandenen Systeme unterstützt wird.

## *3.2. Softwareentwicklungsmethoden*

Das vorangegangene Kapitel 3.1 erläutert die Notwendigkeit einer Methode zur Entwicklung von verteilten Krankenakten sowie die Anforderungen an diese. Die Disziplin, die sich mit Methoden zur praktischen, aber systematisch strukturierten Entwicklung von Software befasst, ist das Software Engineering. Aus diesem Grund stellt die Menge der Methoden des Software Engineerings die Basis für die Auswahl einer geeigneten Entwicklungsmethode zur besonderen Unterstützung der Entwicklung verteilter Krankenakten dar.

Als Ausgangspunkt für die Untersuchung der derzeit bekannten Methoden des Software Engineerings werden die Standardwerke von Sommerville (Sommerville 2007) sowie von Ludewig und Lichter (Ludewig & Lichter 2010) verwendet. Eine weitere Quelle ist die IEEE Publikation „Guide to the Software Engineering Body of Knowledge“ (SWEBOK) (Abran u. a. 2004), die den Stand der Technik in der Disziplin *Software Engineering* im Jahr 2004 wiedergibt. Die zusammenfassende Darstellung der Methoden des Software Engineerings erfolgt in Kapitel 3.2.1. Das Kapitel 3.2.2 enthält eine Gliederung der Bereiche des Software Engineerings, während in Kapitel 3.2.3, auf Basis der ermittelten Anforderungen, die Auswahl einer geeigneten Methode erfolgt.

### 3.2.1. Methoden im Software Engineering

Am Beginn des 3. Kapitels wird der Begriff der Methode als eine wissenschaftliche, planmäßige oder folgerichtige Art des Vorgehens definiert. Zu jeder theoretisch untersuchten oder formal beschriebenen Methode existiert eine zugehörige Methodik. Sie beschreibt den theoretischen Hintergrund und die Anwendungsprinzipien der Methode und ermöglicht so das Erlernen und die Anwendung der Methode. Sommerville beschreibt Methoden des Software Engineering als strukturierte Ansätze für die Softwareentwicklung, deren Ziel es ist, die kostengünstige Herstellung qualitativ hochwertiger Software zu ermöglichen (Sommerville 2007:S. 37). Daher ist jede planmäßige und strukturierte Vorgehens- oder Verfahrensweise, die die Entstehung von Software unterstützt, eine Methode des Software Engineerings. Laut Sommerville (Sommerville 2007:S. 38, Abb. 1.4) besteht eine Methode im Software Engineering aus den folgenden Bestandteilen:

- Beschreibung von Systemmodellen (Modelltypen und Notation)
- Regeln (die für die Systemmodelle gelten)
- Empfehlungen (zur erfolgreichen Anwendung der Methode)
- Anleitung zum Vorgehen.

Ein weiterer wichtiger Bestandteil einer Methode ist ihr Name. Durch den Namen wird eine Methode eindeutig identifizierbar. Je klarer der Name die Methode beschreibt, desto einfacher kann die Methode durch potenzielle Nutzer als Kandidat für die Lösung eines ihrer Entwicklungsprobleme identifiziert werden.

Ludewig und Lichter stellen mit der Aussage „*Im Software Engineering ist die Bedeutung der Modelle noch größer, weil sie nicht* [nur] *Zwischenschritte, sondern Endpunkte unserer Arbeit darstellen: Eine Spezifikation, aber auch ein Programm ist ein Modell“* (Ludewig & Lichter 2010:S. 3) fest, dass das Ziel jeglicher Aktivität im Software Engineering die Erstellung eines Modells ist. Folglich gibt der Bestandteil „Beschreibung von Systemmodellen“ aus der Aufzählung der Bestandteile einer Methode des Software Engineerings nach Sommerville das Ergebnis der Anwendung einer Methode wieder. Das Ergebnis kann dabei sowohl ein abstraktes, im Rahmen der Planung eines Softwaresystems entstandenes Systemmodell als auch ein konkretes Stück ausführbare Software sein.

Ergänzt um diese zusätzlichen Erkenntnisse besteht eine Methode des Software Engineerings aus:

- einem eindeutigen Namen,
- festgelegten Modelltypen als Ergebnis der Anwendung der Methode,
- einer Anleitung zur schrittweisen Anwendung der Methode,
- Regeln und
- Empfehlungen zur erfolgreichen Anwendung der Methode.

Eine Methode kann vollkommen neuartig sein, aber auch aus einer Kombination bereits bekannter Methoden bestehen oder eine bereits bekannte Methode, bezogen auf einen speziellen Kontext oder Aspekt verfeinern.

### 3.2.2. Teilbereiche des Software Engineerings

Die Menge der Methoden des Software Engineerings ist sehr umfangreich. Sie finden in unterschiedlichen Phasen des Entstehungsprozesses von Software ihren Einsatz. Aus diesem Grund ist zur zielgerichteten Betrachtung und Auswahl von Methoden eine systematische Gliederung des Software Engineerings in seine Teilbereiche nötig. Eine gängige Methode zur Gliederung der Teilbereiche des Software Engineerings ist die Gliederung nach Phasen, Abschnitten oder Aufgaben innerhalb eines möglichst allgemeinen Entwicklungsprozesses.

Im Inhaltsverzeichnis des „Guide to the Software Engineering Body of Knowledge (SWEBOK)“ (Abran u. a. 2004) wird Software Engineering in die folgenden Teilbereiche gegliedert:

- Anforderungsanalyse (Software Requirements)
- Softwareentwurf (Software Design)
- Umsetzung oder Programmierung (Software Construction)
- Softwaretests (Software Testing)
- Softwarewartung (Software Maintenance)
- Konfigurationsmanagement (Software Configuration Management)
- Software Engineering Management
- Software Engineering Process
- Qualitätssicherung (Software Quality).

Eine Analyse des Inhalts des Buchs zu Software Engineering von Ludewig und Lichter (Ludewig & Lichter 2010) ergibt die folgenden Aufgabenbereiche, die zu denen aus SWEBOK nahezu identisch sind:

- Planung des Vorgehens
- Analyse der Zieldomäne
- Spezifikation der Anforderungen
- Entwurf der Anwendung
- Codierung oder Umsetzung der Anwendung
- Test der Anwendung
- Dokumentation der Software
- Qualitätssicherung und Bewertung
- Weiterentwicklung und Wartung.

Die Gliederung nach Ludewig und Lichter beinhaltet, mit Ausnahme des Konfigurationsmanagements, alle in SWEBOK gefundenen Kategorien und wird somit für die weitere Betrachtung, ergänzt um den Punkt Konfigurationsmanagement, als Ausgangsbasis für die Gliederung der Teilbereiche des Software Engineerings verwendet. Sommerville (Sommerville 2007:S. 106) untergliedert das Gebiet *Softwareentwurf* in die sechs grundlegenden Entwurfsaktivitäten: Architekturentwurf, abstrakte Spezifikation, Schnittstellenentwurf, Komponentenentwurf, Entwurf der Datenstrukturen und Algorithmenentwurf. Diese Aktivitäten sind sowohl in strukturierten als auch agilen Entwicklungsprozessen zu finden

(Sommerville 2007:S.107, letzter Absatz). Bei genauer Betrachtung zeigt sich, dass der Entwurfsprozess von Sommerville in zwei Phasen zu je drei Entwurfsaktivitäten gegliedert ist.

Die erste der Phasen befasst sich mit Aktivitäten zum abstrakten Entwurf der Softwarearchitektur für das Zielsystem. Sie erstreckt sich über die Aktivitäten *Architekturentwurf*, *abstrakte Spezifikation* und *Schnittstellenentwurf*. Der Architekturentwurf dient der Gliederung des Gesamtsystems in Subsysteme. Im Rahmen der abstrakten Spezifikation werden die Funktionen und Randbedingungen für die einzelnen Subsysteme spezifiziert. Der Schnittstellenentwurf dient der Spezifikation der Schnittstellen zur Kommunikation zwischen den Subsystemen.

Die zweite Phase befasst sich mit dem detaillierten Entwurf des Softwaresystems. Dazu werden in der Aktivität *Komponentenentwurf* die Subsysteme weiter in einzelne Softwarekomponenten untergliedert. Die weiterführenden Aktivitäten zum Entwurf der Datenstrukturen und Algorithmen vervollständigen den detaillierten Entwurf des Softwaresystems.

Alternativ zu der Aufteilung des Software Engineerings in die geschilderten aufgabenbasierten Bereiche kann es auch auf Basis verschiedener Eigenschaften seiner Methoden gegliedert werden. Hierbei ist beispielsweise die Gliederung anhand des Formalisierungsgrads, also in agile Methoden und kontrollierte bzw. strikt strukturierte Entwicklungsprozesse gebräuchlich (Sommerville 2007:S. 430). Eine weitere für die Gliederung gebräuchliche Eigenschaft ist das durch die Methode unterstützte Entwicklungs- oder Programmierparadigma. So lassen sich unter dem Begriff des objektorientierten Software Engineerings (siehe z. B. Jacobson 1997) die Methoden zum Entwurf und zur Umsetzung objektorientierter Anwendungssysteme oder unter dem Begriff des komponentenbasierten Software Engineerings (siehe z. B. Heineman & Councill 2001) die Methoden für Entwurf und Umsetzung komponentenbasierter Anwendungssysteme zusammenfassen.

Zusätzlich können Methoden des Software Engineerings auch nach ihrem Umfang oder Anteil am Software-Entwicklungsprozess gegliedert werden. Es gibt Methoden, die der Strukturierung des gesamten Entwicklungsprozesses dienen. Andere Methoden dienen dagegen der Lösung technischer Detailprobleme oder der systematischen Analyse und Erfüllung von funktionalen oder nichtfunktionalen Anforderungen. Manche Methoden können an unterschiedlichen Stellen im Entwicklungsprozess eingesetzt werden, andere hingegen sind auf einen präzise abgegrenzten Einsatzbereich beschränkt.

Das Software Engineering besitzt folglich neben der häufig gewählten prozessphasen- und aufgabenbasierten Gliederung weitere relevante Gliederungskonzepte, die als zusätzliche Gliederungsdimensionen den Detaillierungsgrad des zentralen Gliederungskonzepts verfeinern können. Bei der Methodenauswahl gilt es zu berücksichtigen, dass es grundsätzlich keine optimale Methode gibt, die für alle Fragestellungen bei allen Softwareprojekten das bestmögliche Ergebnis liefert (Sommerville 2007:S. 38). Die Auswahl einer geeigneten Methode muss immer auf Basis des verfolgten Ziels und der Rahmenbedingungen des zugrunde liegenden Softwareprojekts bzw. der Klasse von Softwareprojekten getroffen

werden. Einige dieser Ziele und Rahmenbedingungen lassen sich direkt mit den dargestellten Gliederungskonzepten in Beziehung setzen. Sie bieten somit die Möglichkeit die Menge der verfügbaren Methoden zu einer Menge der potenziell geeigneten Methoden einzugrenzen.

### 3.2.3. Methodenauswahl

In diesem Kapitel wird auf Basis der Anforderungen an eine geeignete Entwicklungsmethodik (Kapitel 3.1) aus den ermittelten Teilbereichen des Software Engineerings (Kapitel 3.2.2) eine geeignete Methode ausgewählt. Die Methode muss zu Verbesserungen bei der Entwicklung und Evaluation verteilter Krankenakten führen. Der primäre Anwendungsbereich liegt in der Entwicklung der verteilten Krankenakten (siehe S. 102 Absätze 1 und 2). Die Aufgabe der Entwicklung besteht aus den Teilbereichen *Anforderungsanalyse*, *Architekturentwurf*, *Software-Entwurf* und *Umsetzung*. Im Software Engineering sind für die Phase der Entwicklung eine Vielzahl unterschiedlicher Methoden verfügbar.

Die Menge der infrage kommenden Methoden lässt sich anhand der Intensität der Berücksichtigung domänenspezifischer oder technischer Eigenschaften in zwei Gruppen teilen. Die erste Gruppe beinhaltet Methoden, die als primäres Artefakt ein problemadäquates Modell der Zieldomäne mit ihren Bestandteilen und anwendungsrelevanten Zusammenhängen besitzen. Lösungsansätze für technische Probleme werden bei dieser Verfahrensweise erst im Rahmen der nächsten Planungsphase in das Modell integriert. Diese Gruppe wird im weiteren Text als Gruppe der am Domänenmodell orientierten Methoden bezeichnet. Die zweite Gruppe bilden die Methoden, deren erster Entwicklungsschritt die Konzeption eines technischen Modells ist, das in einem zweiten Schritt um die Inhalte der Zieldomäne erweitert wird. Das technische Modell ist dabei häufig ein generisches Modell, das noch unabhängig von den domänenspezifischen Inhalten ist. Diese Gruppe wird im weiteren Text als Gruppe der an technischen Aufgaben des Systems orientierte Methoden bezeichnet.

Um als Basisprinzip für eine geeignete Methode gelten zu können, müssen die Methoden der beiden Gruppen je mindestens die folgenden Anforderungen erfüllen:

- Findet in den Phasen *Analyse* und *Entwicklung* seinen Einsatz und hat Auswirkungen auf die Umsetzung und Evaluation der Systeme.
- Beinhaltet ein Konzept zur einfachen und praktischen Verwendung wissenschaftlicher Erkenntnisse bei der Entwicklung neuer Systeme.
- Verbessert die empirische Untersuchbarkeit der entstandenen Systeme.
- Besteht aus Bestandteilen, deren Qualität schrittweise empirisch nachgewiesen werden kann.
- Berücksichtigt sowohl funktionale als auch nichtfunktionale Anforderungen verteilter Krankenakten.
- Ist mit anderen Methoden des Software Engineerings kombinierbar.
- Ist unabhängig von den Standards aus Kapitel 2.5 anwendbar.

**Domänenmodell-orientierte Methoden**

Das Domänenmodell ist der Hauptbestandteil Domänenmodell-orienierter Entwicklungsansätze. Es beschreibt einen oder mehrere Aspekte der Zieldomäne in einer für das gewählte Programmierparadigma geeigneten Form. Das Prinzip eines objektorientierten Domänenmodells wird beispielsweise durch das Domain-Model-Pattern beschrieben. Es stellt das Domänenmodell als ein Objektmodell dar, das sowohl Daten- als auch Verhaltensaspekte der Zieldomäne beinhaltet (Fowler 2002:S. 116). Betrachtet man die an den verschiedenen Aspekten der medizinischen Arbeitsweise orientierten Ausführungen zum Aufbau medizinischer Informationssystemen durch Haas (Haas 2009:Kapitel 3–8), so müssen Methoden, die das Domänenmodell als zentrales Entwicklungsartefakt betrachten, als möglicherweise geeignet bei der Methodenauswahl berücksichtigt werden.

Ein typischer Vertreter Domänenmodell-orientierter Ansätze ist das Domain driven Design (Evans 2004; Nilsson 2006). Bei diesem Ansatz ist die Abbildung der Bestandteile und Aufgaben der Zieldomäne in einem Domänenmodell die zentrale Aufgabe des Entwurfs. Die Anwendung selbst wird, ausgehend vom Domänenmodell, in einem iterativen, agilen Prozess entwickelt.

Das Domänenmodell hat im Domain driven Design eine klar definierte Menge von Element-Typen wie z. B. Entitäten (Evans 2004:S. 91), Value Objects (Evans 2004:S. 98) oder Services (Evans 2004:S. 105–106), aus denen ein dem Zweck der Softwareentwicklung adäquates Modell der Zieldomäne erstellt wird. Üblicherweise werden solche Modelle mittels grafischer Modellierungssprachen abgebildet (z. B. als UML-Klassendiagramm).

Beim Domain driven Design stehen hauptsächlich inhaltliche Aspekte und davon besonders die Zusammenhänge zwischen den beteiligten Klassen von Objekten im Vordergrund der Betrachtung. Wird Domain driven Design als vorherrschendes Entwicklungsprinzip angewendet, werden inhaltliche Aspekte als wenig volatile Bestandteile identifiziert und relativ starr im Systemmodell verankert. Aus diesem Grund eignet sich Domain driven Design als primäres Prinzip vor allem für Anwendungsbereiche, in denen die zentrale Komplexität der Anwendung nicht in der technischen Fragestellung, sondern in der Komplexität der Anwendungsdomäne selbst steckt (Evans 2004:S. xii).

Die zentralen Herausforderungen bei der Entwicklung verteilter Krankenakten sind allerdings technischer Natur. Der Aspekt der Verteilung beinhaltet primär die Integrations- und Kommunikationsfähigkeit der Systeme in einem Verbund von heterogenen Systemen. Dabei wird eine Vielzahl von konkreten Domänenmodellen für verschiedene jeweils abgegrenzte medizinische Anwendungsdomänen verwendet. Die Methoden zu deren Modellierung sind durch die in Kapitel 2.5 (vor allem 2.5.2 und 2.5.4) beschriebenen Standards standardisiert. Eine zentrale Forderung dieser Arbeit ist jedoch die Unabhängigkeit von den einzelnen Standards (siehe S. 103 u. 104) und dadurch auch eine Unabhängigkeit von konkreten Datenmodellen. Deshalb muss die Erstellung eines Domänenmodells zwar Bestandteil der Methode zur Entwicklung verteilter Krankenakten, aber nicht deren dominierendes Entwicklungsprinzip sein.

Fowler führt im Vorwort zu Evans Buch über das Domain driven Design (Evans 2004:S. xviii) an, dass der größte Mehrwert des Domain driven Design darin liegt, die Kommunikation zwischen den Domänenexperten und den Softwareentwicklern zu verbessern. In Kapitel 3.1.2 wird eine Verbesserung der Kommunikation zwischen Entwicklern und Forschern gefordert. Daraus resultiert die Frage, ob Domain driven Design auch dazu geeignet ist, diesen Kommunikationsweg zu verbessern.

Aus der verbesserten Kommunikation zwischen Entwicklern und Forschern soll eine schnellere Adaption wissenschaftlicher Erkenntnisse durch die Praxis sowie eine verbesserte empirische Untersuchbarkeit von entwickelten Systemen resultieren. Ziel der Untersuchungen ist dabei die Identifikation von Zusammenhängen zwischen der Verwendung softwaretechnischer Konzepte und deren Auswirkung auf die Nutzbarkeit und Eignung des Systems für seinen praktischen und dauerhaften Einsatz. Für eine solche Untersuchung ist die Abstraktion von softwaretechnischen Konzepten zu übersichtlichen, kombinierbaren und benennbaren Einheiten nötig.

Durch die Anwendung von Domain driven Design wird jedes Modell zu einem individuell an den Anforderungen des einzelnen Anwendungsfalls ausgerichteten Modell. Ohne den Einsatz zusätzlicher Methoden sind keine abstrakten Modellbestandteile identifizierbar oder benennbar. Aus diesem Grund können die Bestandteile des Modells nicht aufgrund der Anwendung einer durch das Domänenmodell getriebenen Methode, sondern nur aufgrund der Anwendung dritter Methoden wiedererkennbar benannt und dadurch empirisch untersucht werden. Im Rahmen der Definition der Anforderungen an eine geeignete Entwicklungsmethodik wird der schrittweise durchzuführende empirische Nachweis für die Qualität der Bestandteile der verwendeten Methode gefordert. Diese Anforderung kann nur durch die zusätzliche Verwendung von Architektur- und Entwurfsmustern (Patterns) erfüllt werden. Dass sich Domain driven Design und Patterns dazu eignen, gemeinsam verwendet zu werden, zeigt Nilsson (Nilsson 2006) durch seine Methode zur Entwicklung von Softwaresystemen, die Domain driven Design und Patterns kombiniert.

Zusammenfassend kann festgehalten werden, dass Domänenmodell-getriebene Ansätze grundsätzlich zur Entwicklung verschiedenartiger medizinischer Informationssysteme geeignet sind. Die Komplexität bei der Entwicklung verteilter Krankenakten resultiert allerdings aus den notwendigen Kommunikations-, Integrations- und Sicherheitsmechanismen. Außerdem ist eine hohe Flexibilität bei der Gestaltung von Datenstrukturen und Prozessen zur Abbildung der fortlaufend veränderlichen medizinischen Dokumentation erforderlich. Zudem verbessert die Anwendung eines solchen Ansatzes die empirische Untersuchbarkeit nicht in der gewünschten Form. Deshalb ist eine Verwendung als primäres Prinzip ausgeschlossen, Domain driven Design kann somit nicht als geeigneter Kandidat für das der Entwicklungsmethode primär zugrunde liegende Prinzip betrachtet werden. Eine aufgabenbezogene Integration von Ideen des Domain driven Designs als untergeordnetes Prinzip ist jedoch vorteilhaft. Sie kann z. B. durch die Integration des Domänenmodell-Musters in die Entwicklungsmethode für verteilte Krankenakten erreicht werden.

**An technischen Aufgaben des Systems orientierte Methoden**

Methoden, die ihren primären Fokus nicht auf die fachlich-inhaltlichen Aspekte der zu entwickelnden Systeme richten, sondern vorrangig technische Probleme lösen, um innerhalb der entstandenen Software-Architektur in geeigneter Weise die fachlich-inhaltlichen Aspekte der Anwendung umsetzen zu können, unterscheiden sich substanziell von den bisher vorgestellten Domänenmodell-basierten Ansätzen. Sie widersprechen ihnen allerdings nicht. Die Notwendigkeit ihres Einsatzes beruht meist auf der Existenz technischer Problemstellungen, die auf fachliche Anforderungen funktionaler und vor allem nichtfunktionaler Art zurückzuführen sind. Die aus dem Aspekt der Verteilung resultierenden Anforderungen (S. 103) an Sicherheit und Zuverlässigkeit sowie die Forderung nach flexibler standardübergreifender Integrationsfähigkeit sind solche technischen Probleme. Auch die Forderung nach Generizität und konfigurativer Veränderbarkeit bei der Abbildung von Datenmodellen (siehe Kapitel 2.5) und Prozessen (siehe S. 104 und Kapitel 2.3) ist in die Rubrik technischer Problemstellungen einzuordnen. Aus diesen Gründen müssen Methoden, die zur Lösung technischer Aufgaben dienen, als möglicherweise geeignete Methoden zur Entwicklung verteilter Krankenakten angesehen werden.

Methoden, die auf der Anwendung von Patterns (Mustern) basieren, sind typische Vertreter zur Lösung softwaretechnischer Probleme geeigneter Methoden. Dabei sind Patterns abstrakte Lösungsmuster für wiederkehrende Probleme, die durch Konkretisierung in verschiedenen realen Problemsituationen angewendet werden. Durch ihre abstrakte Darstellungsform beschreiben sie Lösungsmuster, die noch unabhängig von konkreten Implementierungstechnologien und Standards sind.

Patterns haben eindeutige Namen (Buschmann u. a. 1996:S. 19) und machen so wiederkehrende technische Lösungsmuster in verschiedenen Systemen identifizier- und benennbar. Existieren synonyme Patterns, so wird die Eindeutigkeit des zugrunde liegenden Konzepts durch die Auflistung bekannter Synonyme oder verwandter Patterns im Rahmen der Beschreibung der Patterns hergestellt. Entsprechende Angaben enthalten beispielsweise die Beschreibungsschablonen von Gamma u. a. sowie von Buschmann u. a. (Buschmann u. a. 1996:S. 20–21; Gamma u. a. 2010:S. 6).

Die Eignung von Patterns zur Dokumentation von Softwarearchitekturen wird unter anderem von Prechelt u. a. (Prechelt u. a. 2002) bestätigt. Auch die Aussage von Malich, *„Pattern erweitern somit das Entwurfsvokabular eines Designers bzw. Entwicklers und können deshalb auch innerhalb der Dokumentation einer Softwarearchitektur* […] *verwendet werden“* (Malich 2008:S. 107), unterstreicht diese Eignung. Eine Studie von Unger und Tichy (W. F. Tichy & Unger 2000) belegt, dass der Einsatz von Patterns die Kommunikation von Architektur- und Design-Aspekten eines Softwaresystems vereinfacht. Die Verwendung von Patterns und Pattern-Kombinationen in Systemen kann, gemeinsam mit qualitativen Informationen über die Nutzbarkeit der Systeme publiziert werden. Sammlungen dieser Daten können als Basis für die Durchführung empirischer Untersuchungen über die Zusammenhänge zwischen erkennbaren Nutzbarkeitseigenschaften und den Entscheidungen bezüglich Architektur und Design des Anwendungssystems eingesetzt werden. Dadurch verbessert die Verwendung von Patterns die nachträgliche gemeinsame Untersuchbarkeit

größerer Mengen in ihrer Art ähnlichen IT-Systemen. Zudem können durch empirische Untersuchungen auch die qualitativen Eigenschaften der einzelnen Patterns als Bestandteile der Entwicklungsmethodik ermittelt werden.

Patterns werden als Architektur- (Ludewig & Lichter 2010:S. 429) und Entwurfsmuster (Ludewig & Lichter 2010:S. 437) bei der Entwicklung von Softwaresystemen eingesetzt. Sie dienen dazu, erprobte Prinzipien zur Lösung von Problemen in die praktische Entwicklung einfließen zu lassen (Buschmann u. a. 1996:S. 5). Dabei können Patterns auch wissenschaftliche Erkenntnisse, z. B. aus Untersuchungen an Mengen existierender Systeme, in anwendungsfreundlicher Form dokumentieren. Auf diese Weise werden wissenschaftliche Erkenntnisse direkt – in einer von Softwareentwicklern häufig genutzten Form – in die Praxis der Softwareentwicklung integriert.

Sollen auf Patterns basierende Methoden als geeignet für die Entwicklung verteilter Krankenakten gelten, müssen sie (siehe S. 110) mit anderen Methoden des Software Engineerings und mit beliebigen Vorgehensmodellen (siehe S. 105) kombinierbar sein. Grundsätzlich sind Patterns für die verschiedenen Phasen des Entwicklungsprozesses als Architektur- oder Entwurfsmuster sowie als besonders feingranulare Idiome (Buschmann u. a. 1996:S. 14) verfügbar. Die vielfältige Kombinierbarkeit lässt sich gut am Beispiel einer Kombination mit dem vorab betrachteten Domain driven Design und dem Domain-Model-Pattern beschreiben.

Wie bereits am Ende der Ausführungen zu Domänenmodel basierten Ansätzen erwähnt, sind Patterns und Domain driven Design keine einander grundsätzlich widersprechenden Methoden. Das beweist auch das Buch „Applying Domain-Driven Design and Patterns“ von Jimmy Nilsson (Nilsson 2006), das den Einsatz von Patterns im Rahmen des Domain Driven Designs beschreibt. Darin werden Domain driven Design als vorherrschender und die Anwendung von Patterns als untergeordneter Methodenbestandteil betrachtet. Fowler zeigt durch die Eingliederung des Domänenmodells als Domain-Model-Pattern (Fowler 2002:S. 117), dass auch der umgekehrte Ansatz, das Domänenmodell als spezielles Architekturmuster zu betrachten, valide ist. Dadurch wird es möglich, die zentrale Idee aus dem Domain driven Design in beliebige patternbasierte Ansätze zu integrieren.

Patternbasierte Methoden erfüllen folglich alle sieben auf S. 110 gestellten Anforderungen für die Auswahl eines geeigneten Basisprinzips. Durch die große Vielfalt und unterschiedliche Granularität der verfügbaren Patterns verbleibt den Entwicklern verteilter Krankenakten ein hohes Maß an Flexibilität bei der Auswahl ihres Vorgehensmodells und zusätzlicher Entwicklungsmethoden. Die Entscheidung zur Auswahl patternbasierter Methoden als Basisprinzip für die Entwicklung von verteilten Krankenakten wird zusätzlich durch bestehende Projekte, die sowohl bei der Entwicklung als auch zur Dokumentation des Projekts Patterns verwendet haben, unterstützt.

Ein gutes Beispiel hierfür ist die Publikation von Hasselbring und Zietsche (Hasselbring & Ziesche 1997), die die Architektur eines Systems zur Replikation von Patientendaten zwischen heterogenen Subsystemen innerhalb eines verteilten Krankenhausinformationssystems auf Basis der dabei benutzten Design-Patterns beschreibt. Die Verwendung der Design-

Patterns in der Architekturbeschreibung gewährt dem Leser der Publikation einen sehr detaillierten und dennoch verständlichen Überblick über die technischen Details der Softwarearchitektur des vorgestellten Systems.

Ein weiteres Beispiel der praktischen Anwendung von Mustern in medizinischen Informationssystemen sind die in openEHR beschriebenen Archetypen. Sie stellen als medizinischen Konzepten oder Begriffen zugeordnete Datenmodell-Bestandteile feingranulare Muster niedriger Abstraktionsebene, vergleichbar mit Idiomen, dar.

Die beiden Beispiele zeigen, dass Patterns bereits in verschiedenen Ansätzen zu medizinischen Informationssystemen eingesetzt werden. Das Potenzial, das hinter der Idee der Patterns und der auf der Anwendung von Patterns basierenden Entwicklungsansätze steckt, wird allerdings noch nicht in ausreichendem Umfang ausgeschöpft. Dazu wird eine umfassende Sammlung von, für die Entwicklung verteilter Krankenakten relevanten Patterns und eine dazugehörige Methodik zur geführten Anwendung und Kombination der gesammelten Patterns benötigt.

## 3.3. Softwareentwicklung mit Patterns

Infolge der Auswahl des Pattern-Konzepts als zentralen Bestandteil der Entwicklungsmethodik für verteilte Krankenakten ist es nötig die verfügbaren Arten von Patterns, die Darstellungsformen von Pattern-Sammlungen sowie den Begriff *Pattern* selbst detailliert zu betrachten, denn eine unsystematische Sammlung von Patterns kann die in Kapitel 3.1 beschriebenen Anforderungen nicht erfüllen. Dazu ist vielmehr eine umfassend strukturierte Sammlung aufzubauen, die neben den Patterns auch weitere Informationen, z. B. zur Kombinierbarkeit der Patterns enthält.

Naheliegende Darstellungsformen für strukturierte Pattern-Sammlungen sind die Pattern-Story und die Pattern-Sprache (Pattern Language). Beide stellen Patterns im Bezug zu anderen Patterns dar, sind dabei allerdings unterschiedlich umfassend.

### 3.3.1. Patterns

Patterns sind in Kapitel 3.2.3 als Kernbestandteil der Entwicklungsmethodik ausgewählt worden. Deshalb ist eine detaillierte Betrachtung des Begriffs und seiner Verwendung in verschiedenen Disziplinen notwendig. Grundsätzlich entspricht der englischsprachige Begriff *Pattern* dem deutschen Begriff *Muster*. Seine Bedeutung reicht von einem Muster im Sinne einer Vorlage (z. B. Musterbrief oder Musterlösung) bis hin zu optisch erkennbaren Strukturen (beispielsweise auf Textilien) die ebenfalls als Muster bezeichnet werden.

Innerhalb der Informatik wird der Begriff des Musters in zwei Teilgebieten verstärkt verwendet. Zum einen wird in den Fachgebieten Künstliche Intelligenz (KI) und Bildverarbeitung unter dem Begriff der Mustererkennung (pattern recognition) die Erkennung von Mustern in Mengen von Daten mittels spezifischer Algorithmen oder Methoden der KI verstanden. Zum anderen werden im Software Engineering unter dem Begriff *Pattern* Mengen verschiedener Arten von Lösungsmustern zusammengefasst. Diese Art der Verwendung des Begriffes *Pattern* geht auf den Architekturwissenschaftler Christopher

Alexander zurück (Ludewig & Lichter 2010:S. 429), der in seinen Werken „The Timeless Way of Building“ (Alexander 1979) und „A Pattern Language“ (Alexander u. a. 1977) die Nutzung von Patterns im Rahmen der Entwicklung und Konzeption von Städten und Gebäuden beschreibt. Dadurch legt er die Grundlagen für die in dieser Arbeit relevante Verwendung des Begriffs *Pattern* fest. Er definiert ein Pattern als eine dreiteilige Regel, die eine Beziehung zwischen einem bestimmten Kontext, einem Problem und einer zugehörigen Lösung herstellt (Alexander 1979:S. 247). Eine wichtige Eigenschaft jedes Patterns ist, dass seine Beschreibung ausreichend abstrakt sein muss, um die enthaltene Lösung durch Konkretisierung flexibel auf einen Bereich ähnlicher Probleme anwenden zu können.

Seit Anfang der 1990er-Jahre ist dieses Konzept von verschiedenen Informatikern (siehe Ludewig & Lichter 2010:S. 429, 430 u. 437) angepasst und in die Informatik übernommen worden. Heute sind in der Informatik Patterns für verschiedene Anwendungsbereiche verfügbar. Für das Software Engineering besonders relevante Gruppen von Patterns sind:

- Design-Patterns (siehe z. B. Gamma u. a. 2010)
- Softwarearchitektur-Patterns (siehe z. B. Buschmann u. a. 1996)
- Idiome (siehe z. B. Coplien 1991)
- UI-Patterns (siehe z. B. Ratzka 2010) bzw. Usability Patterns (siehe z. B. Graham 2003)
- Anti-Patterns (siehe z. B. W. J. Brown u. a. 1998).

Die Beschreibung der Patterns durch Alexander ist, neben der bereits erwähnten Dreiteilung in Kontext, Problem und Lösung wenig systematisiert und ausschließlich in textueller Form abgefasst. Für den textuellen Inhalt liefert Alexander vielfältige Anmerkungen und Empfehlungen. Diese Empfehlungen zur internen Strukturierung der textuellen Darstellung sind für die Beschreibung von Patterns in der Informatik nicht geeignet (vgl. Buschmann u. a. 1996:S. xii). Aus diesem Grund beinhalten die diversen Werke zu Patterns in der Softwareentwicklung (z. B. Gamma u. a. 2010; Buschmann u. a. 1996) stärker strukturierte, für alle Patterns eines Katalogs systematisch gleichartig aufgebaute Beschreibungen, die in Form einer für den jeweiligen Katalog gültigen Schablone (siehe z. B. Buschmann u. a. 1996:S. 20–21; Gamma u. a. 2010:S. 6–7) definiert sind. Als zusätzlichen formalen Beschreibungsaspekt enthalten die Schablonen jeweils einen Abschnitt für die grafische Darstellung der Lösung, z. B. in Form von UML-Diagrammen.

Für die Darstellung von Patterns wird in dieser Arbeit die folgende in Anlehnung an Buschmann u. a. (Buschmann u. a. 1996) und Gamma u. a. (Gamma u. a. 2010) gestaltete Schablone verwendet.

- Name
- Kontext (Buschmann) oder Intension (Gamma)
- Problem
- Beziehungen
    - Synonyme Patterns
    - Verwandte Patterns
    - Anderweitig abhängige Patterns

- Lösung
    - Textuelle Beschreibung
    - Strukturelle Darstellung mittels Diagrammen
- Konsequenzen
- Einflüsse
- Ergebnis
- Beispiele.

Abbildung 22: Beschreibungsschablone für Patterns

Mithilfe dieser Schablone ist eine an der Anwendung des jeweils beschriebenen Patterns orientierte Darstellung möglich. Die Punkte *Name*, *Intension* und *Problem* beschreiben das Pattern so, dass es problembezogen ausgewählt werden kann. Im Punkt *Beziehungen* werden Abhängigkeiten zu anderen Patterns dargestellt. Der Hinweis auf synonyme und verwandte Patterns hilft bei der Auswahl, unter einer Menge ähnlicher Patterns das tatsächlich optimal geeignete zu finden, während die Beziehungen zu anderweitig abhängigen Patterns dazu dienen, Patterns zu ermitteln, die Probleme lösen, die aus der Anwendung des aktuell gewählten Patterns resultieren. Der Rest der Beschreibung dient der Erläuterung der Lösung sowie ihrer Konkretisierung hin zu einer konkreten Implementierung. Das Vorgehen und empfehlenswerte Regeln beim Verfassen von Patterns sowie inhaltliche Zusammenhänge zwischen den Bestandteilen der Pattern-Beschreibung werden von Meszdaros und Doble detailliert in ihrer Pattern Language for Pattern Writing (Meszaros & Doble 1997) vorgestellt.

Der zentrale Nutzen von Patterns liegt in der Wiederverwendung bereits bekannten Wissens. Durch die gewählte Beschreibungsform wird es auch für Personen anwendbar, die dieses Wissen nicht selbst generiert haben oder zu ihrem aktiven Wissensschatz zählen (Ludewig & Lichter 2010:S. 429). Auf diese Weise wird es auch unerfahrenen Entwicklern ermöglicht, hochwertige Architektur- und Entwurfslösungen in ihren Softwareprodukten umzusetzen (Gamma u. a. 2010:S. 1). Zudem erlaubt die Darstellungsform der Patterns auch eine langfristige Konservierung von Architektur- und Entwurfswissen und macht durch ihre abstrakte Darstellungsweise Lösungen, die für eine bestimmte Plattform erarbeitet wurden, auch auf der Basis verwandter Technologien anwendbar. Außerdem eignen sich Patterns auch zur Dokumentation der Architektur und des Designs bereits entwickelter Softwaresysteme (Buschmann u. a. 1996:S. 6) und verbessern dadurch deren Wartbarkeit (Prechelt u. a. 2001).

Im Bereich der Design-Patterns existieren neben der pragmatischeren, an der Anwendung- und den Anwendungsergebnissen der Patterns orientierten Betrachtungsweisen auch verschiedene Ansätze zur mathematisch-formalen Beschreibung von Patterns und Pattern-Systemen. Ein Beispiel dafür ist das Rho-Kalkül (J. M. Smith & Stotts 2002) von Jason McColm Smith und David Stotts. Es basiert auf dem für die Abbildung objektorientierter Systeme verwendeten Sigma-Kalkül (Abadi & Cardelli 1996:S. 60 ff.) und der Zerlegung von Design-Patterns in sogenannte elementare Design-Patterns (siehe ebenfalls J. M. Smith & Stotts 2002). Elementare Design-Patterns sind kleine Muster, die die abstrakten Grundkonstrukte objektorientiert entwickelter Software als Muster abbilden. Sie können durch Kombinationen untereinander und mit bereits gebildeten Design-Patterns sämtliche höherwertigen, also komplexeren Design-Patterns abbilden.

Die Qualität der formalen Darstellung mittels des Rho-Kalküls zeigt sich z. B. in einem Experiment, das aufbauend auf verschiedenen mittels Rho-Kalkül beschriebenen Patterns eine automatisierte Identifikation dieser Muster in objektorientiertem Quellcode ermöglicht (J. M. Smith & Stotts 2003). Derart formal beschriebene Patterns können unter Beibehaltung ihrer Eigenschaften umgeformt werden. Diese Eigenschaft erlaubt es, unterschiedliche, aber gleichbedeutende Implementierungen dem jeweils richtigen Pattern zuzuordnen.

Patterns sind folglich abstrakte Lösungsmuster, die in unterschiedlichen Formen für unterschiedliche Verwendungszwecke dargestellt werden können. Einerseits geschieht das durch mittels Schablonen normierte textuelle Darstellungsformen, die der praktischen Anwendung der Patterns entgegenkommt. Andererseits existieren für den Teilbereich der Design-Patterns auch Möglichkeiten, diese formal darzustellen. Diese formale Form der Darstellung kann beispielsweise genutzt werden, um in verschiedenen Untersuchungsszenarien die Eindeutigkeit bezüglich der Zuordnung konkreter Implementierungen zu Patterns zu gewährleisten.

### 3.3.2. Pattern-Story und Pattern-Sequenz

Bei der praktischen Verwendung von Patterns wird schnell erkennbar, dass Patterns nicht voneinander unabhängig existieren (Buschmann u. a. 1996:S. xii). Werden mehrere Muster in einem Stück Software verwendet, so beeinflussen sie sich gegenseitig entweder positiv oder negativ. Dieses Verhalten resultiert aus der Tatsache, dass die Anwendung eines Musters zwar ein konkretes Problem löst, meist aber auch, bezogen auf einen anderen Aspekt, ein oder mehrere weitere Probleme sichtbar macht. Diese Probleme oder Defizite können durch die Anwendung weiterer Patterns gelöst werden. Dieses Vorgehen kann fortgesetzt werden, bis keine für den Anwendungsbereich relevanten resultierenden Probleme mehr entdeckt werden. Das Ergebnis dieses Verfahrens ist ein gerichteter Graph, der ausgehend von einem oder mehreren initial gewählten Patterns, über mehrere Stufen auf eine Menge weiterer Patterns verzweigt. Eine solche nach den Regeln einer Pattern-Sprache erstellte Aneinanderreihung von Patterns nennt man Pattern-Sequenz (Henney u. a. 2005:S. 26; Buschmann u. a. 2007b:S. 191).

Pattern-Stories (Buschmann u. a. 2007b:S. 185) sind in Form einer Geschichte anwendungsbezogen dargestellte Pattern-Sequenzen (Henney u. a. 2005:S. 25). Sie beschreiben die Anwendung einer Pattern-Sequenz in einem festgelegten praktischen Kontext (Henney u. a. 2005:S. 27). Dieser Kontext kann z. B. die Konzeption eines fiktiven oder real existierenden Softwaresystems sein.

Zur Beschreibung von Pattern-Stories und Pattern-Sequenzen sind verschiedene Methoden verfügbar. Für die textuell erzählende Beschreibungsform werden neben der klassisch sequenziell lesbaren und bereits durch Christopher Alexander verwendeten Beschreibungsform inzwischen auch andere Formen vorgeschlagen, die durch Stellen, die vom Leser eine explizite Auswahl des weiteren Textflusses erfordern, stärker interaktiv gestaltet sind. Eine solche Darstellungsform ist die Beschreibung von Pattern-Sequenzen in Form von interaktiven Pattern-Stories (Siddle & Platts 2009; Siddle 2011).

Eine grafische Darstellungsform, die sich aus Alexanders Abbildung des Beziehungssystems von Teilen seiner Pattern-Sprache ableitet, ist die Darstellung der Sequenz in Form eines

Graphen aus mit Pfeilen verbundenen Knoten (Abbildung 23). Dabei sind Notationsformen mit (z. B. Schumacher u. a. 2005:S. 72) und ohne (z. B. Hanmer 2007:S. 35) beschriftete Pfeile gebräuchlich.

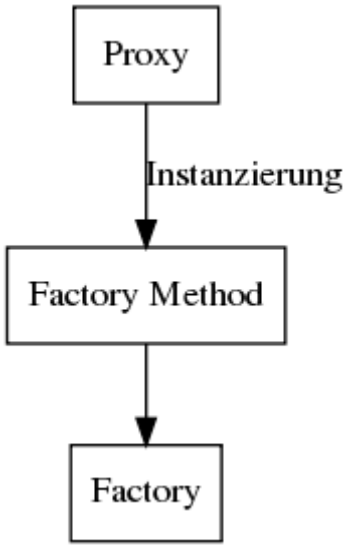



Abbildung 23: Pattern-Sequenz als gerichteter Graph

Zur formalen und zugleich illustrativen Beschreibung der einer Pattern-Story zugrunde liegenden Pattern-Sequenz kann mit der Pattern Instance Notation (PIN) (J. M. Smith 2011) auf eine geeignete, aussagekräftigere grafische Notationsform zugrückgegriffen werden.

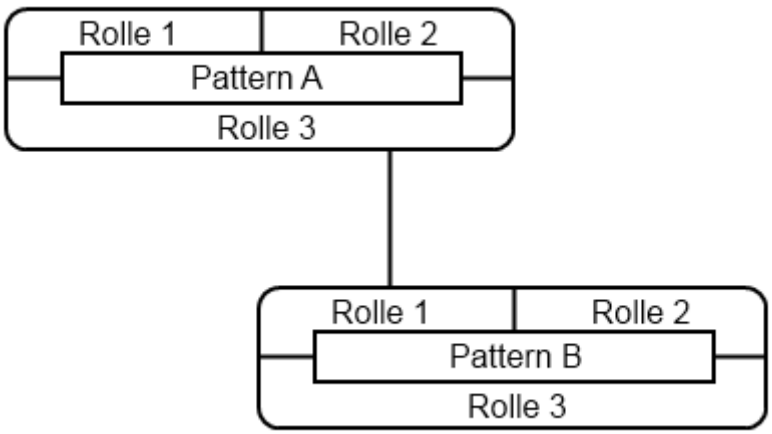



Abbildung 24: Standard-PIN-Diagramm mit verbundenen Rollen (vgl. J. M. Smith 2011)

Abbildung 24 zeigt die Abbildung einer Kombination zweier Patterns mittels PIN. Darin ist erkennbar, dass durch die verwendete Notation neben der Tatsache der Kombination der Patterns, in Form der Angabe der Rollen, die die Patterns im Rahmen der Verbindung einnehmen eine weitergehende, die Kombination exakter spezifizierende Aussage getroffen wird.

Textuelle Beschreibungen von Pattern-Sequenzen in der Form von Pattern-Stories eignen sich sehr gut zur Erläuterung einer Kombination von Patterns, da in ihnen auf Probleme, Empfehlungen aber auch auf grundlegende Intentionen eingegangen werden kann. Zugleich besteht durch ihre sehr umfangreiche Textform eine große Schwierigkeit darin, die zugrunde

liegende Pattern-Sequenz zu jeder Zeit für den Leser präsent zu halten. Um die Übersichtlichkeit einer Pattern-Story zu verbessern, kann diese z. B. durch ein PIN-Diagramm erweitert werden, das schon vor dem Lesen der textuellen Beschreibung einen Überblick über die Struktur der enthaltenen Pattern-Sequenz vermittelt.

### 3.3.3. Pattern-Sprache

Eine Pattern-Sprache (Pattern Language) beinhaltet eine Vielzahl verschiedener Patterns, einschließlich ihrer Beziehungen, die gleichsam als Kombinationsregeln verstanden werden dürfen. Aus einer Pattern-Sprache lässt sich eine Vielzahl von Pattern-Sequenzen und Pattern-Stories bilden. Der Begriff der Pattern-Sprache wurde erstmals 1977 durch Christopher Alexander u. a. in der Monografie „A Pattern Language“ (Alexander u. a. 1977) eingeführt und durch das 1979 herausgegebene Werk „The Timeless Way of Building“ (Alexander 1979) präzisiert. Alexanders Pattern-Sprache ist eine systematisierte Sammlung von Lösungsmustern für die Architektur und den Städtebau, einschließlich einer Vielzahl zugehöriger und explizit benannter Kombinationsbeziehungen. Sie dient als Darstellungsform dazu, die Verwendung von Patterns durch erleichtern des Auffindens geeigneter bzw. einem Problem angepasster Kombinationen zu unterstützen.

Patterns enthalten in abstrakter Form Wissen über die Gestaltung von Lösungen zu konkreten Problemen. Pattern-Sprachen dienen dazu, dieses Wissen gemeinsam mit zusätzlichem Wissen über die Kombinierbarkeit der enthaltenen Patterns strukturiert und wiederverwendbar darzustellen. Dazu bieten sie, im Sinne einer durch neue Erkenntnisse stetig weiterentwickelbaren Grammatik, Regeln zur Kombinierbarkeit von Patterns. Diese Regeln werden durch Beziehungen zwischen den Patterns repräsentiert. In jeder Beziehung nehmen die beteiligten Patterns konkrete Rollen (J. M. Smith 2011) zur Erfüllung des durch die Kombination der Patterns zu erreichenden Ziels ein. Das Konzept *Pattern-Sprache* zur Beschreibung komplexer Pattern-Systeme ist 1977 gemeinsam mit dem des Patterns veröffentlicht worden. Es handelt sich demnach um inhärent zusammengehörige Konzepte, denn Patterns und Pattern-Sprachen entfalten nur in Kombination ihr volles Potenzial (Buschmann u. a. 2007b:S. 245).

Buschmann, Henney und Schmidt definieren eine Pattern-Sprache, bezogen auf ihren Anwendungsbereich in der Informatik, als ein Netzwerk eng verwobener Muster, das gleichzeitig einen Prozess für die systematische Lösung einer Menge von einander verwandten Problemen der Softwareentwicklung beschreibt (Buschmann u. a. 2007b:S. 260). Dabei kann die Verwandtschaft der Probleme beispielsweise auch durch eine gemeinsame Anwendungsdomäne erfüllt werden (Buschmann u. a. 2007b:S. 250). Eine reine Sammlung von Patterns, die keine konkreten Regeln für die kombinierte Verwendung seiner Patterns enthält, ist laut Buschmann u. a. nicht geeignet, da als wesentlicher Bestandteil für eine Entwicklungsmethodik eine konkretisierbare Vorgehensweise zur problembezogenen Verwendung der Bestandteile der Sammlung fehlt (Buschmann u. a. 2007b:S. 250).

Berücksichtigt man die in Kapitel 3.1 gestellten Anforderungen an eine Entwicklungsmethodik für verteilte Krankenakten und die Tatsache, dass in Kapitel 3.2.3 die Wahl auf die Verwendung eines auf Patterns basierenden Ansatzes gefallen ist, so können die durch die ausschließliche Anwendung von Patterns nicht lösbaren Anforderungsbestandteile zu großen

Teilen durch die Darstellung der domänenspezifischen Patternsammlung in Form einer Pattern-Sprache erfüllt werden.

Der Aufbau einer Pattern-Sprache für die Domäne *verteilte Krankenakten* wird durch zusätzliche Bedingungen eingeschränkt. Christopher Alexander stellt in „The Timeless Way of Building“ konkrete Anforderungen, die ein Pattern-System bezüglich seiner Vollständigkeit erfüllen muss, um als Pattern-Sprache bezeichnet werden zu können. Eine Pattern-Sprache muss sowohl morphologisch als auch funktional vollständig sein (Alexander 1979:S. 316). Die morphologische Vollständigkeit ist erfüllt, wenn ein Zielsystem durch die Pattern-Sprache für eine Umsetzung ausreichend eindeutig und konkret beschrieben werden kann (Alexander 1979:S. 316). Die funktionale Vollständigkeit ist dann gegeben, wenn die Pattern-Sprache es erlaubt, alle inneren Abhängigkeiten vollständig aufzulösen (Alexander 1979:S. 317). Das bedeutet, dass für ein Pattern, das in der Sprache enthalten ist, alle Patterns, von denen es selbst abhängig ist, enthalten sein müssen. Außerdem ist eine Pattern-Sprache nur dann vollständig, wenn alle enthaltenen Patterns vollständig beschrieben sind (Alexander 1979:S. 318).

Vor allem die funktionale Vollständigkeit der Pattern-Sprache ist schwer zu beweisen, da existierende Abhängigkeiten gegebenenfalls noch unbekannt sein können. Auch die Frage nach einer ausreichend eindeutigen Beschreibung, wie sie bezüglich der morphologischen Vollständigkeit gestellt wird, ist kaum eindeutig beweisbar und somit nicht positiv zu beantworten. Da aus diesen Gründen die Präsentation einer erwiesenermaßen vollständigen Pattern-Sprache für die Entwicklung verteilter Krankenakten nicht möglich ist, wird in den folgenden Kapiteln deren Strukturkonzept und Kernbestandteil vorgestellt. Unter dem Begriff *Kernbestandteil der Pattern-Sprache für verteilte Krankenakten* ist eine erkennbar nicht vollständige Pattern-Sprache, die aber alle in den Kapiteln 2 und 3 als wichtig identifizierten Bereiche mindestens grundlegend abdeckt, zu verstehen.

# 4. Strukturkonzept der Pattern-Sprache

In diesem Kapitel wird das für die Darstellung der Pattern-Sprache gewählte Strukturkonzept erläutert. Das Strukturkonzept ist von verschiedenen Faktoren beeinflusst. Einerseits bestimmen bestehende allgemeine Konzepte zur Strukturierung von Pattern-Sprachen als bekannte Prinzipien die theoretische Basis, während andererseits die Anforderungen der Domäne verteilte Krankenakten die konkrete Umsetzung und Ausprägung des Konzepts festlegen.

Der Begriff der Struktur ist bezogen auf Pattern-Sprachen sehr umfassend, da er sowohl die durch die Beziehungen zwischen Patterns inhärent bestehenden Strukturen als auch die absichtlich vom Autor der Pattern-Sprache gewählten Gliederungseinheiten beinhaltet. Während die Beziehungsstruktur sich automatisch aus der Menge der gewählten Patterns ergibt und die Kombinierbarkeit von Patterns dokumentiert, dienen die zusätzlich vom Autor festgelegten Strukturen dazu, die Pattern-Sprache übersichtlich zu gestalten, um deren Verwendbarkeit zu verbessern.

Die für die Strukturierung einer Pattern-Sprache für verteilte Krankenakten gewählten Klassen von Strukturelementen und deren Beziehungen werden im weiteren Text als Metamodell der Pattern-Sprache bezeichnet. Die Kapitel 4.2.1 bis 4.2.5 erläutern die enthaltenen Klassen näher. Das Metamodell wird in Form eines Klassendiagramms dargestellt und zusätzlich textuell beschrieben.

## *4.1. Auswahl des Strukturkonzepts*

Die Auswahl von geeigneten Bestandteilen für die Gestaltung eines Konzepts zur Strukturierung der Pattern-Sprache für verteilte Krankenakten erfolgt auf Basis bekannter Methoden. Sie werden abhängig von den für den speziellen Anwendungsfall geltenden Rahmenbedingungen ausgewählt.

Durch die Betrachtung verschiedener publizierter Pattern-Sprachen (Alexander u. a. 1977; Buschmann u. a. 2007a usw.), aber auch weniger vollständiger Pattern-Sammlungen (Gamma u. a. 2010; Buschmann u. a. 1996 usw.) lässt sich feststellen, dass diese jeweils in strukturierter textueller Form wiedergegeben werden. Inhaltlich Zusammengehöriges, weil fachlich Verwandtes wird beispielsweise in gemeinsamen Kapiteln dargestellt. Till Schümmer hat die diversen Gliederungs- und Strukturierungskonzepte in einem Aufsatz (Schuemmer 2003:S. 1) systematisch betrachtet und zusammengefasst. Er identifiziert die folgenden vier Ansätze:

- Lineare Klassifikation von Patterns in Kapiteln oder Familien
- Hierarchische Klassifikation von Patterns
- Netzwerk von Patterns
- Eine einzelne oder eine Sammlung von Pattern-Sequenzen.

Bei seinen weiteren Betrachtungen stellt Schümmer fest, dass bei Sammlungen, die eine große Zahl von Patterns beinhalten, häufig eine hierarchische Struktur verwendet wird (Schuemmer 2003:S. 2). Der erarbeitete Kernbestandteil der Pattern-Sprache für verteilte Krankenakten (siehe Kapitel 5) umfasst eine Sammlung von rund 220 Patterns. Aus diesem Grund ist die hierarchische Struktur ein aussichtsreicher Kandidat für die Auswahl einer geeigneten Grundstruktur. Buschmann, Henney und Schmidt beschreiben in ihrem Buch „On Patterns and Pattern Languages“ (Buschmann u. a. 2007b) die Verwendung von Eigenschaften bzw. Anforderungen der Zieldomäne (Buschmann u. a. 2007b:S. 219) oder von zugrunde liegenden Absichten bzw. Zielsetzungen (Buschmann u. a. 2007b:S. 221) als mögliche Basis für den Aufbau einer Klassifikation der Patterns einer Pattern-Sprache. Das Ziel der zu erarbeitenden Pattern-Sprache ist die Vereinfachung der Entwicklung von verteilten Krankenakten. Dabei handelt es sich um eine konkrete Anwendungsdomäne. Deshalb erscheint die Strukturierung nach Eigenschaften, die mindestens einen Bezug zu dieser Anwendungsdomäne besitzen, sinnvoll. Die Kombination aus diesen beiden vorgestellten Ansätzen ist eine hierarchische, nach Eigenschaften mit Relevanz für die Zieldomäne gegliederte Struktur. Sie muss aufgrund der genannten Annahmen für die Strukturierung der umfangreichen und domänenspezifischen Pattern-Sprache für verteilte Krankenakten geeignet sein.

Salingaros erklärt in seinem Aufsatz „The Structure of Pattern Languages“ (Salingaros 2000), dass Muster innerhalb der Sprache zusätzlich in verschiedenen Ebenen (Level) existieren. Er bezieht sich damit unter anderem auf die bereits von Alexander (Alexander 1979:S. 312 ff.) festgestellte Tatsache, dass umfassendere Muster aus je einer Menge von kleineren Mustern bestehen. Betrachtet man diese Beziehung (A besteht aus B) grundlegender, so ist festzustellen, dass sie durch die Aussage „A führt zur Anwendung von B“ allgemeiner und ebenfalls treffend beschreibbar ist. Denn wenn ein Pattern höherer Ebene zur Anwendung kommt, führt das dazu, dass automatisch auch die untergeordneten Muster, aus denen es besteht, verwendet werden müssen.

Für die Entwicklung von Softwaresystemen gibt es Patterns für viele verschiedene Bereiche. Sie existieren unter anderem für verschiedene Phasen des Entwicklungsprozesses, so z. B. für die Anforderungsanalyse (Withall 2007) oder für die Konzeption einer Softwarearchitektur (Fowler 2002) und deren Umsetzung im Software-Design (Gamma u. a. 2010). Dabei führt die Anwendung von Mustern der Anforderungsanalyse als Resultat zur Anwendung von Softwarearchitektur-Mustern, da die ermittelten Anforderungen Entscheidungsgrundlage für die Auswahl der entsprechenden Architekturmuster sind. Die Architekturmuster wiederum werden durch die Anwendung jeweils einer Kombination von Design-Patterns umgesetzt. Dabei wirken Anforderungsanalyse, Softwarearchitektur und Design als logische Ebenen, die die Menge der enthaltenen Muster zusätzlich zu der gewählten domänenbezogen-hierarchischen Darstellung gliedern.

Die Darstellung der Pattern-Sprache erfolgt in zwei verschiedenen Formen. Textuell als fortlaufende Auflistung der enthaltenen Muster, einschließlich ihrer jeweiligen Beziehungen, und als gerichteter Graph mit den in 4.2 beschriebenen Kanten- und Knotentypen. Die graphische Darstellung der Pattern-Sprache und ihrer Teile erfolgt in Anlehnung an die Konzepte von Christopher Alexander (Alexander 1979:S. 313, Abbildungen), die Pattern Instance Notation (J. M. Smith 2011) sowie die von Schobert (Schobert 2005) verwendeten

Darstellung. Alexanders ursprünglicher Visualisierungsansatz beinhaltet die Verwendung eines gerichteten Graphen ohne beschriftete Kanten. Dabei haben die Kanten die Bedeutung, dass das Objekt, von dem die Kante ausgeht, das andere Objekt als Bestandteil benötigt. In der Pattern-Instance Notation werden die Beziehungen durch die Angabe der Rollen, die die Patterns im Rahmen der Beziehung einnehmen, um zusätzliche Bedeutungsinformation erweitert. Schobert (Schobert 2005:S. 37) beschreibt die farbige Markierung von Pattern-Gruppen. In Anlehnung daran wird eine Kennzeichnung von Gruppen zusammengehöriger Patterns durch zusätzliche Knoten und Kanten anstelle der farbigen Markierung im hier vorliegenden Konzept integriert. Eine farbige Kennzeichnung der Gruppen ist aufgrund der großen Zahl der ermittelten Gruppen und der endlichen Zahl gut unterscheidbarer Farben auszuschließen.

## 4.2. Bestandteile des Strukturkonzepts

Wie in Kapitel 4.1 im Rahmen der Auswahl des Strukturkonzepts erläutert, wird die Pattern-Sprache durch in diesem Kapitel näher zu erläuternde Strukturobjekte in eine hierarchische Struktur gegliedert. Eine weitere Strukturierung wird durch die Zuordnung der Elemente zu verschiedenen Ebenen erreicht. Zweck dieses Kapitels ist es, die Bestandteile dieses Konzepts jeweils einzeln bezüglich ihrer Verwendung und Bedeutung zu erläutern. In Kapitel 4.3 werden die verschiedenen Bestandteile zueinander in Beziehung gesetzt und zu einem Modell zusammengefasst.

### 4.2.1. Patterns

Patterns sind die eigentlichen Hauptbestandteile einer Pattern-Sprache. Sie bilden gleichsam das Vokabular der Sprache und werden durch die in den Punkten 4.2.2 bis 4.2.5 beschriebenen Elemente logisch gruppiert und zueinander in Beziehung gesetzt. Inhaltlich entsprechen sie dem Pattern-Begriff, der in Kapitel 3.3.1 erläutert ist.

Im Zuge der Abbildung von Ausschnitten der Pattern-Sprache in Form von Graphen werden Patterns in dieser Arbeit als Knoten mit rechteckigem Rahmen dargestellt. Die Knoten enthalten in der ersten Zeile den Namen des Patterns und in der zweiten Zeile die Zeichenfolge <<Muster>> um sie eindeutig als Patterns, zu Deutsch „Muster“, erkennbar zu machen.

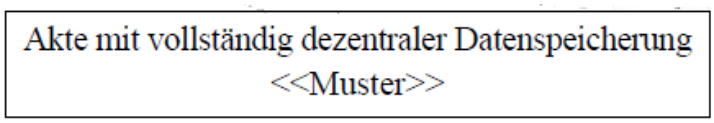



Abbildung 25: Darstellung eines Pattern-Knotens

In der textuellen Darstellungsform wird in dieser Arbeit zwischen neuen Patterns, die spezifisch für verteilte Krankenakten vorgestellt werden, und solchen, die bereits weithin geläufig sind, aber auch für die Entwicklung verteilter Krankenakten benötigt werden, unterschieden.

Die Beschreibung bekannter Patterns ist verglichen mit der Beschreibung der speziellen Patterns deutlich verkürzt. Sie besteht aus dem Namen des Patterns, der Zuordnung des

Patterns zu einer Ebene (4.2.3) sowie einem oder mehreren Schwerpunkten (4.2.2) und einer oder mehreren Gruppen (4.2.4). Die Angaben zum Inhalt des Patterns werden durch einen oder mehrere Verweise auf Quellen, die das Pattern beschreiben, und eine Kurzbeschreibung ersetzt. Darauf folgt die Abbildung der im Rahmen dieser Arbeit durch Literaturrecherche und praktische Versuche ermittelten direkten Beziehungen des Musters in Form eines gerichteten Graphen. Unter dem Graphen folgt eine Auflistung der im Graphen gezeigten Beziehungen, einschließlich je einer kurzen textuellen Erläuterung zu jeder Beziehung.

Die Beschreibung der Patterns, die für verteilte Krankenakten spezifisch sind, ist umfangreicher aufgebaut. Sie umfasst die bei bekannten Patterns verwendeten Elemente für Ebene, Schwerpunkt(e) und Gruppen sowie eine auf die Elemente *Kontext*, *Problem*, *Lösung*, *Beispiel* sowie *Beziehungen* reduzierte Beschreibung gemäß der in Abbildung 22 dargestellten Pattern-Schablone.

### 4.2.2. Schwerpunkt

Schwerpunkte sind die gröbste Gliederungsebene des Strukturkonzepts. Sie fassen Mengen von Patterns und Gruppen zusammen, die gemeinsam zur Erreichung einer bestimmten Eigenschaft (z. B. der Zuverlässigkeit des Systems) dienen. Der Bezeichner *Schwerpunkt* wurde für diese Gliederungseinheiten gewählt, weil die entsprechenden Eigenschaften im Rahmen der Ermittlung übergeordneter Anforderungen (siehe S. 103, Auflistung von Gruppen nichtfunktionaler Anforderungen) als Schwerpunkte in den verschiedenen identifizierten Anforderungen zeigen. Ein Schwerpunkt beschreibt somit auch eine Gruppe zusammengehöriger funktionaler oder nichtfunktionaler Anforderungen bzw. Eigenschaften.

Die Auswahl der Schwerpunkte ist ausschließlich durch die aus den speziellen Erfordernissen der Domäne resultierenden Anforderungen bestimmt. Das bedeutet, dass die Anforderungen, die für alle Systeme zur Abbildung verteilter Krankenakten gemeinsam gelten, die Basis für die Auswahl der Schwerpunkte bilden. Bei der Menge der Schwerpunkte handelt es sich allerdings nicht um eine abgeschlossene Sammlung. Technische Neuerungen, Veränderungen der rechtlichen Rahmenbedingungen, neue Erkenntnisse über die Zieldomäne und andere grundlegende Veränderungen innerhalb der Zieldomäne können grundsätzlich zusätzliche Schwerpunkte notwendig machen oder die Menge um nicht mehr sinnvolle Schwerpunkte reduzieren.

Durch die in Kapitel 2.4 (insbesondere 2.4.2, S. 63) genannten Anforderungen werden für verteilte Krankenakten die folgenden vier Schwerpunkte als besonders relevant identifiziert und als Basis für den Aufbau der Pattern-Sprache für verteilte Krankenakten verwendet.

- **Flexibilität** (bestehend aus Erweiterbarkeit, Veränderbarkeit und Anpassbarkeit)
- **Kommunikation**
- **Sicherheit**
- **Zuverlässigkeit.**

Abbildung 26: Auflistung der gewählten Schwerpunkte

In grafischen Darstellungen werden Schwerpunkte innerhalb dieser Arbeit als Knoten mit einer ovalen Umrandung in der ersten Ebene des Graphen abgebildet. Bei der Darstellung mehrerer Schwerpunkte in einem Graphen werden sie als Ovale dargestellt, die in der zweiten Ebene direkt mit Kanten zum Wurzelknoten des Graphen verbunden sind. Innerhalb des Ovals wird der Name des Schwerpunkts zwischen spitze Klammern (z. B. <Zuverlässigkeit>) gestellt.

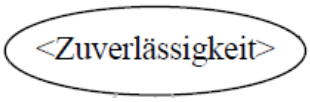



Abbildung 27: Darstellung eines Schwerpunkt-Knotens

Textuell beschrieben werden die Schwerpunkte in Kapitel 5 in gewöhnlichem Fließtext. Zudem enthält die Beschreibung eine Auflistung und detaillierte Beschreibung aller enthaltenen Gruppen. Die Beschreibung einer jeden Gruppe wird durch einen Graphen vervollständigt, der die Abhängigkeiten zwischen den enthaltenen Patterns und Gruppen zeigt.

### 4.2.3. Ebene

Nikos A. Salingaros beschreibt in seinem Aufsatz „The Structure of Pattern Languages“ (Salingaros 2000), dass Muster innerhalb der Sprache in verschiedenen Ebenen (Level) existieren. Eine grundlegende Idee, die in der vorliegenden Arbeit dazu verwendet wird, um die Patterns in drei Ebenen zu gliedern, die die Phasen eines stark verallgemeinerten Software-Entwicklungsprozesses widerspiegeln. Im Unterschied zur Gliederung nach Schwerpunkten ist hier die Granularität der Patterns sowie die gemeinsame Art des Einsatzes für eine Zuordnung zu einer Ebene ausschlaggebend.

Das Ebenenkonzept der vorgestellten Pattern-Sprache umfasst die folgenden drei Ebenen:

- Anforderungsanalyse-Patterns
- Architektur-Patterns
- Design-Patterns.

Prinzipiell kann es innerhalb der Pattern-Sprache auch Muster geben, die keiner Ebene eindeutig zuordenbar sind. Diese Muster können als keiner Ebene zugeordnete Objekte trotzdem in allen anderen Gliederungselementen verwendet werden. Daher ist die Zuordnung eines Objekts zu einer Ebene nur dann angebracht, wenn sie eindeutig durchgeführt werden kann. Das Konzept der Ebenen ist somit in der vorgestellten Pattern-Sprache ein nicht vollständiges Konzept, das ausschließlich dazu dient, die Übersichtlichkeit der Sammlung zu verbessern.

### 4.2.4. Gruppe

Zur Erreichung einer aufgabenorientierten Struktur werden die Patterns innerhalb der Schwerpunkte in Gruppen von Patterns nach der Ähnlichkeit der Probleme, die sie lösen, oder nach der Ähnlichkeit des Lösungsansatzes gegliedert.

Häufig gibt es verschiedene Patterns, die gleiche Probleme auf unterschiedliche Weise oder unterschiedliche Probleme auf ähnliche Weise lösen. Diese Muster weisen eine besondere inhaltliche Nähe zueinander auf. Sie besitzen oftmals ähnliche Beziehungen zu anderen Patterns oder zu Gruppen von anderen Patterns. Bei der Auswahl eines Patterns zur Lösung eines vorliegenden Problems gilt es, aus einer Menge geeigneter Patterns den geeigneten Kandidaten für den konkreten Einsatzzweck zu ermitteln. Dazu ist es hilfreich, Patterns die einander bezogen auf ein bestimmtes Kriterium ähnlich sind, in Gruppen mit sprechenden Namen zusammenzufassen. Jedes Pattern kann Mitglied in mehreren Gruppen sein, da es hinsichtlich anderer Eigenschaften Ähnlichkeiten zu unterschiedlichen Patterns aufweisen kann.

Die Idee, Patterns logisch zu gruppieren, ist nicht neu und wird in verschiedenen Quellen in unterschiedlichen Kontexten erwähnt. Beispiele hierfür sind die Bände 1 und 5 der Reihe „Pattern-Oriented Software Architecture“ (Buschmann u. a. 1996:S. xiii; Buschmann u. a. 2007b:S. 342). Auf S. 342 in Band 5 sprechen die Autoren von „group of Patterns“, in Band 1 hingegen werden die Gruppen als „families of related patterns“ bezeichnet. Die Bezeichnung „family of patterns“ wird auch bei Gamma et al. (Gamma u. a. 2010:S. 22) verwendet, während Salingaros (Salingaros 2000) die Bezeichnung „pattern groups“ wählt.

Die Begriffe *Familie* und *Gruppe* werden häufig für die Bezeichnung der Gruppierungseinheit von inhaltlich zusammengehörigen Patterns verwendet. Der Begriff der Familie führt durch seine Verwechslungsmöglichkeit mit dem Begriff der Familie aus der Mathematik möglicherweise zu unerwünschten bzw. falschen Annahmen über die Eigenschaften einer Familie bzw. Gruppe von Patterns. Aus diesem Grund wird in der vorliegenden Arbeit der Begriff *Gruppe* verwendet.

Die Zuordnung von Patterns zu Gruppen ist eine Vereinfachungsform der Abbildung von Ähnlichkeitsbeziehungen bezüglich bestimmter Eigenschaften, die eigentlich zwischen allen möglichen Zweier-Tupeln von Patterns innerhalb der Gruppe existieren. Drei der von Schümmer (Schuemmer 2003:S. 2) identifizierten Beziehungstypen, nämlich

- X has a similar problem as Y,
- X and Y are members of the same family,
- X and Y can be found in the same known use,

bilden Beziehungen ab, die durch eine Gruppierung von Patterns vereinfacht dargestellt werden können.

Da sich die Pattern-Gruppen nicht direkt aus den Kapiteln 2 und 3 ableiten wird hier am Beispiel der Gruppe "Ermittlung der Flexibilitätsanforderungen" aus der Ebene *Anforderungsanalyse* des Schwerpunkts *Flexibilität* – der zweiten in Kapitel 5 beschriebenen Gruppe – erläutert, wie die in Kapitel 5 vorgestellten Gruppen gebildet wurden. Grundsätzlich ist der Prozess für jede Gruppe sehr individuell und orientiert sich ausschließlich an den Rahmenbedingungen, die aus dem Zweck der Gruppe resultieren. Allen Gruppen ist aber gemein, dass sie gebildet wurden, um entweder eine konzeptionelle Lücke im bestehenden Pattern-System zu schließen oder um bereits aus anderen Gründen im Pattern-System vorhandene Patterns

aufgrund sonst nicht berücksichtigter gemeinsamer Eigenschaften zu gruppieren. Die Gruppe "Ermittlung der Flexibilitätsanforderungen" wurde beispielsweise gebildet, um eine Lücke zwischen den Ergebnissen des Patterns "Veränderlichkeitsanalyse" und den Patterns der Ebene *Architektur* im Schwerpunkt *Flexibilität* zu schließen. Wie auf S. 108 gezeigt, teilen Ludewig und Lichter die Aufgaben der Ebene *Anforderungsanalyse* in die Abschnitte "Analyse der Zieldomäne" und "Spezifikation der Anforderungen". Das Pattern "Veränderlichkeitsanalyse" dient der Ermittlung von wahrscheinlich veränderlichen Rahmenbedingungen und Systembestandteilen und ist somit klar der "Analyse der Zieldomäne" zuzuordnen. Für die Gestaltung der Softwarearchitektur sind allerdings klar formulierte Anforderungen bezüglich der unterschiedlichen Aspekte der Veränderlichkeit erforderlich. Deshalb wird eine anfangs noch leere Gruppe "Ermittlung der Flexibilitäts-anforderungen" definiert um Patterns zusammenzufassen, die den Prozess der Ermittlung und Definition solcher Anforderungen unterstützen. Im Anschluss daran wird sowohl mittels Literaturrecherche als auch durch Abgleich mit allen bereits im Pattern-System vorhandenen Patterns nach jenen Patterns gesucht, die aufgrund ihres Zwecks in diese Gruppe eingeordnet werden können. Zusätzlich muss für jedes neu hinzugefügte Pattern geprüft werden, ob es in einem anderen Kontext nicht auch in weitere bereits bestehende Gruppen eingeordnet oder aufgrund fehlender Relevanz für verteilte Krankenakten ausgeschlossen werden muss.

In der grafischen Darstellung von Ausschnitten der Pattern-Sprache werden Gruppen in Form eines Knotens mit ovalem Rahmen dargestellt. Der Knoten wird mit dem Namen der Gruppe beschriftet und enthält keine weiteren Inhalte (siehe Abbildung 28).

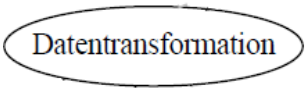



Abbildung 28: Darstellung eines Gruppen-Knotens

### 4.2.5. Beziehungstypen

Die Beziehungen zwischen den Patterns sind ein ebenso wichtiger Bestandteil einer Pattern-Sprache wie die Patterns selbst. Diese Tatsache wurde bereits durch Christopher Alexander in „The Timeless Way of Building“ (Alexander 1979:S. 314, kursiver Text) festgehalten.

Schümmer identifiziert zehn verschiedene Formen von Beziehungen (Schuemmer 2003:S. 2) zwischen Patterns. Drei dieser Formen sind bereits im Kapitel 4.2.4 der Bildung von Pattern-Gruppen zugeordnet worden. Somit verbleiben 7 Typen von Beziehungen die direkt zwischen den Patterns angewendet werden:

- X verwendet Y in seiner Lösung
- X ist eine Variante des Patterns Y
- X spezialisiert Y im Sinne von Vererbung
- X ist nicht näher spezifiziert durch Verwendung im Beschreibungstext von Pattern Y abhängig
- X ist im Rahmen einer Pattern-Sequenz mit Y verbunden
- X erwähnt Y in seiner Kontextbeschreibung
- Die Kombination aus X und Y führt zur Notwendigkeit der Verwendung von P.

Van Welie und van der Veer (Van Welie & Van der Veer 2003) identifizieren für ihre Pattern-Sprache für Interaktionsdesign drei etwas allgemeiner gehaltene Gruppen von Beziehungen (Übersetzung von Schobert 2005:S. 10 übernommen):

- Zusammenfassung (Aggregation)
- Spezialisierung
- Assoziation.

Beeinflusst von den vorgestellten Beziehungstypen sind für die Abbildung der Pattern-Sprache für verteilte Krankenakten die folgenden fünf Typen definiert:

**A führt zur Anwendung von B:** Durch die Verwendung eines Pattern A wird die Verwendung eines Pattern B wahrscheinlich; beispielsweise, weil das Ergebnis der Anwendung von A eine Voraussetzung schafft, die die Anwendung von Pattern B erst ermöglicht, oder aber weil die Anwendung von Pattern A ein Problem a löst, aber ein Problem b schafft, das erst durch die Verwendung von Pattern B wieder gelöst werden kann.

Veränderte Bedeutung dieses Beziehungstyps bei Beziehungen zwischen Patterns und Gruppen:

- ***A ist eine Gruppe:*** Wenn A eine Gruppe ist, führt die Anwendung von mindestens eines ihrer Patterns zur Anwendung von B. Das bedeutet, die Beziehung gilt für jedes Pattern, das in der Gruppe enthalten ist, und wird zur Vermeidung von unnötig vielen Kanten im Graphen zentral der Gruppe angefügt.
- ***B ist eine Gruppe:*** Wenn B eine Gruppe ist, führt die Anwendung von A zur Auswahl von mindestens einem Pattern der Gruppe B oder stellt die zur Auswahl notwendige Information zur Verfügung.

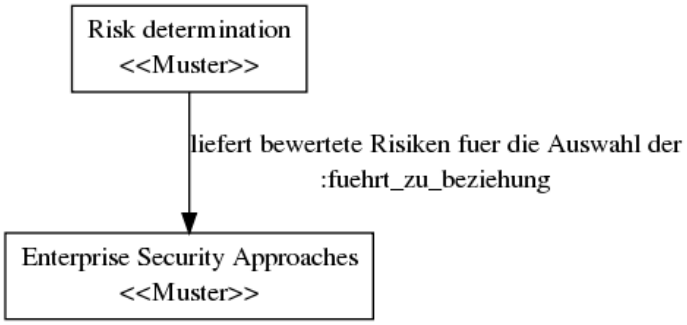



Abbildung 29: Führt-zu-Beziehung

Bei der Verwendung einer Führt-zu-Beziehung ist die Angabe einer die Beziehung näher spezifizierenden Zeichenkette empfehlenswert, da nur so die genaue Bedeutung der Beziehung, also z. B. eine Bedingung, unter der A zu B führt, festgehalten wird.

**A ist vorteilhaft mit B kombinierbar:** Dieser Beziehungstyp ist eine abgeschwächte Form der Führt-zu-Beziehung. Er signalisiert, dass die Kombination zweier Patterns zu vorteilhaften Eigenschaften des Gesamtsystems führt. Die Vorteilhaft-kombinierbar-Beziehung wird ausschließlich zwischen Patterns verwendet und verhält sich ansonsten gleich der Führt-zu-

Beziehung. Auch in der grafischen Darstellung unterscheiden sich die beiden Beziehungstypen allein durch den an der Kante angebrachten Label, der statt mit „führt_zu_beziehung“ mit „vorteilhaft_kombinierbar“ beschriftet ist.

**A ist eine Spezialform von B:** Dieser Beziehungstyp wird verwendet um zu zeigen, dass ein Pattern B eine bezüglich spezifischerer Anforderungen weiterentwickelte Spezialform des Pattern A ist. Es ähnelt in vielfacher Weise der objektorientierten Vererbung und wird aus diesem Grund mit dem in UML für die Abbildung der Vererbung gewählten Pfeilsymbol dargestellt. Dieser Typ wird in der weiteren Arbeit „Spezialisierungsbeziehung“ genannt.

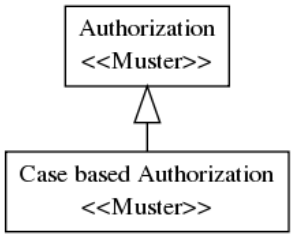



Abbildung 30: Spezialisierungsbeziehung

Spezialisierungsbeziehungen beinhalten in ihrer grafischen Darstellung im Gegensatz zu den bereits vorgestellten Beziehungstypen, kein die Kante zusätzlich beschreibendes Label. Spezialisierungsbeziehungen werden nur zwischen zwei Objekten gleichen Typs verwendet.

**A ist Mitglied der Gruppe B:** Die in Kapitel 5 beschriebenen Gruppen beinhalten Patterns sowie gegebenenfalls auch andere Gruppen. Der Beziehungstyp „A ist Mitglied der Gruppe B“, auch Gruppen-Beziehung genannt, beschreibt die Mitgliedschaft des Elements A (Pattern oder Gruppe) in einer Gruppe B. In der grafischen Darstellung zeigt der Pfeil von der Gruppe B zum Element A.

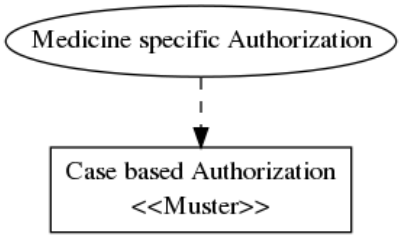



Abbildung 31: Gruppen-Beziehung

**Gruppe A wird in Schwerpunkt B verwendet:** Der Zusammenhang zwischen Gruppen und ihrer Verwendung in den Schwerpunkten wird durch die Schwerpunkt-Beziehung abgebildet. Sie wird ausschließlich zwischen Schwerpunkten und Gruppen oder gruppenlosen Patterns hergestellt.


Abbildung 32: Schwerpunkt-Beziehung

## 4.3. Metamodell

Kapitel 4.2 beschreibt die einzelnen Bestandteile des Strukturkonzepts und benennt bereits einzelne Abhängigkeiten zwischen den verschiedenen Objekttypen. Aus den Objekttypen und ihren Beziehungen lässt sich ein Metamodell entwickeln, das die enthaltenen Abhängigkeiten formal beschreibt und eine softwaretechnische Umsetzung erleichtert. Abbildung 33 zeigt das Metamodell der Pattern-Sprache als Klassendiagramm.

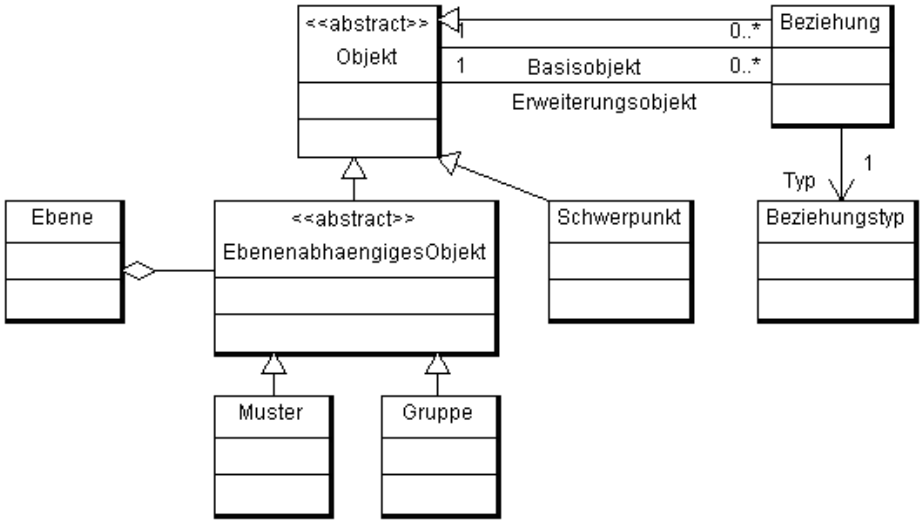



Abbildung 33: Metamodell der Pattern-Sprache

Alle Gliederungsbestandteile, außer den Ebenen, leiten sich von der Klasse *Objekt* ab. Diese Generalisierung ermöglicht es, beliebige Beziehungen zwischen den verschiedenen konkreten Ausprägungen der Klasse *Objekt* mittels einer gemeinsamen Klasse *Beziehung* abzubilden. Eine Beziehung setzt genau zwei Objekte zueinander in Beziehung. Jede Beziehung hat genau einen festgelegten Typ, der ihre Bedeutung beeinflusst. Zusätzlich zu ihrem Typ besitzt die Beziehung noch ein Textfeld, das ihre Bedeutung präzisiert. Die Menge der für eine Pattern-Sprache verwendbaren Beziehungstypen ist in diesem Modell frei definierbar. Auch eine Beziehung selbst ist ein Objekt und kann prinzipiell mit anderen Objekten durch ein zusätzliches Beziehungsobjekt eine typisierte Beziehung eingehen.

Das Metamodell zeigt die Klasse *Schwerpunkt* als einen nicht ebenenabhängigen Typ. Folglich werden Schwerpunkte über die verschiedenen Ebenen hinweg verwendet. Muster (Patterns) und Gruppen hingegen werden, wenn möglich, einer Ebene zugeordnet. Sie sind ebenenabhängige Objekttypen. Die Anwendung des hier vorgestellten Metamodells führt näherungsweise zu dem folgenden schematischen Aufbau von Pattern-Graphen:

| | Schwerpunkt 1 | Schwerpunkt 2 | Schwerpunkt 3 |
|---|---|---|---|
| Ebene 1 | | | |
| Ebene 2 | | | |
| Ebene 3 | | | |
| Ohne Ebene | | | |

Abbildung 34: Schematischer Aufbau - Grobdarstellung

Während sich die Inhalte der Schwerpunkte teilweise überlappen, da Gruppen und Patterns prinzipiell in mehreren Schwerpunkten vorkommen können, ist die Abgrenzung der Ebenen scharf und eindeutig, da jedes ebenenabhängige Objekt nur genau einer oder keiner Ebene zugeordnet sein kann. Durch eine dem logischen Fluss des Entwicklungsprozesses beziehungsweise dem Aufbau von Mustern aus untergeordneten Mustern folgende Wahl der Ebenen, bilden sich unter den Schwerpunkten problembezogen lesbare, ineinander übergehende, baumartige gerichtete Graphen aus Patterns und Gruppen. Diese Graphen können von oben nach unten gelesen und verwendet werden.

Eine für die Entwicklung umfassender Pattern-Systeme wesentliche Eigenschaft dieses Modells ist, dass für die verschiedenen Kombinationen aus je einem Schwerpunkt und einer Ebene – verglichen mit einem Gesamtgraphen – deutlich übersichtlichere thematisch zusammengehörige Teilgraphen generiert werden können. Während der Katalogisierung von mehr als 200 Patterns und über 1000 Beziehungen hat sich die Fähigkeit, das gesamte Pattern-System in thematisch zusammenhängende Subsysteme zergliedern zu können, als für die Beherrschbarkeit der Sammlung notwendig erwiesen.

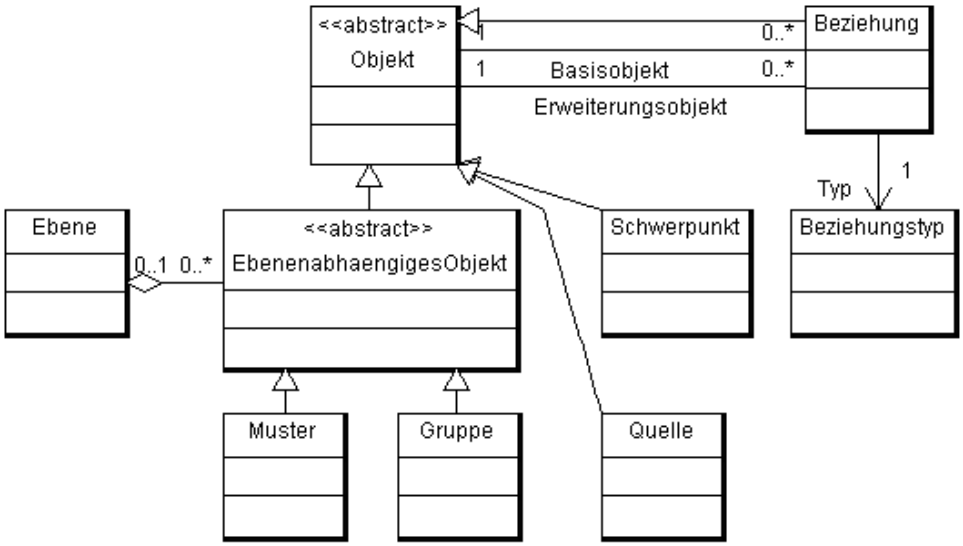



Abbildung 35: Metamodell - erweitert um Quellenangaben

Ein umfassendes Pattern-System besteht neben einer gewissen Anzahl an problembezogenen, neuen Patterns vor allem aus einer großen Zahl bereits bekannter Patterns, zu denen ein umfangreicher Fundus an existierenden Publikationen vorliegt. Diese Publikationen beinhalten zumeist sowohl Relevantes zum Pattern selbst als auch Erläuterungen zu vorliegenden Beziehungen. Quellen sind somit neben den bisher genannten Objekttypen des Metamodells als zusätzlicher relevanter Typ zu integrieren. Abbildung 35 zeigt eine Möglichkeit, Quellen in das vorgestellte Metamodell zu integrieren. Bei dieser Form der Integration kann die Beziehung zwischen dem Objekt (Muster, Beziehung, Gruppe usw.) und seiner Quelle mittels eines zusätzlichen Beziehungstyps, der z. B. Herkunftsbeziehung oder Quellbeziehung heißen kann, erreicht werden.

# 5. Kernbestandteil der Pattern-Sprache

Die bisherigen Kapitel begründen die Eignung einer domänenspezifischen Pattern-Sprache als Grundlage für die systematische Entwicklung von Softwaresystemen zur Umsetzung verteilter Krankenakten und stellen ein Konzept zur Strukturierung einer solchen Pattern-Sprache vor. Eine Pattern-Sprache ist eine Sammlung von Patterns, die sowohl aus Patterns als auch aus den Beziehungen zwischen diesen Patterns besteht. Seit der ursprünglichen Definition des Begriffs durch Christopher Alexander gilt auch, dass es sich bei einer Pattern-Sprache um eine morphologisch und funktional vollständige Sammlung (siehe S. 121) von Pattern und Beziehungen handeln muss. Eine solche Vollständigkeit ist bei der Sammlung von Patterns zu einer umfassenden Anwendungsdomäne, bestehend aus einer Vielzahl verschiedener Problemfelder, schwer zu erreichen oder zu beweisen. Aus diesem Grund wird in diesem Kapitel eine nach den Regeln einer Pattern-Sprache, aber ohne Anspruch auf Vollständigkeit aufgebaute Patternsammlung als Kernbestandteil der Pattern-Sprache für die Entwicklung verteilter Krankenakten vorgestellt.

Die Darstellung der Patternsammlung ist in erster Ebene nach den alphabetisch sortierten Schwerpunkten *Flexibilität*, *Kommunikation*, *Sicherheit* und *Zuverlässigkeit* (vgl. 4.2.2) gegliedert. In ihrer zweiten Ebene erfolgt die Gliederung nach den Ebenen *Anforderungsanalyse*, *Architektur* und *Design* (vgl. 4.2.3). Die Ebenenzuordnung wird auf Basis der Gruppen umgesetzt, d. h. die Patterns werden unabhängig von ihrer eigenen Zuordnung zu einer Ebene, innerhalb der Ebene dargestellt, der die jeweils betrachtete, dem Pattern übergeordnete Gruppe zugeordnet ist.

Die Zuordnung von Patterns und Gruppen zu den Ebenen *Architektur* und *Design* ist zwar sehr anschaulich, in ihren Grenzbereichen jedoch nicht eindeutig. So definieren Buschmann u. a. in POSA4 (Buschmann u. a. 2007a:S. 554 und 557) die beiden Begriffe wie folgt:

Architektur-Pattern: *„An architectural pattern expresses a fundamental structural organization schema for software systems. It provides a set of predefined subsystems, specifies their responsibilities, and incudes rules and guidelines for organizing the relationships between them“* (Buschmann u. a. 2007a:S. 554).

Design-Pattern: *„A design pattern provides a scheme for refining elements of a software system or the relationships between them. It describes a commonly recurring structure of interacting roles that solves a general design problem within a particular context“* (Buschmann u. a. 2007a:S. 557).

Nach diesen beiden Definitionen unterscheiden sich die Ebenen *Architektur* und *Design* lediglich aufgrund der Granularität der Bestandteile eines Softwaresystems, die sie betreffen. Während Architekturpatterns Systeme in Subsysteme gliedern und Regeln für die Organisation der Beziehungen zur Verfügung stellen, befassen sich Design-Patterns mit der Gestaltung von Elementen eines Softwaresystems oder den Beziehungen zwischen diesen. Die Begriffe *Subsystem* und *Element* sind bezogen auf ein Softwaresystem nicht eindeutig gegeneinander abgegrenzt, da beide Teilmengen des Softwaresystems sind.

Der Sinn der Trennung zwischen diesen beiden Ebenen liegt folglich nicht in einer eindeutigen Trennung zweier Mengen, sondern in der anwendungsbezogenen Gliederung des Katalogs. Sie folgt der logischen Abfolge, in der die Patterns bei der Konzeption eines Softwaresystems Anwendung finden, und dient so der praktischen Anwendbarkeit.

Bei der Benennung der Patterns werden nicht durchgängig deutschsprachige Bezeichner verwendet, da sich bei vielen der bekannten Patterns auch im deutschsprachigen Raum die Verwendung der englischen Bezeichner in der Alltagssprache der Softwareentwickler durchgesetzt hat. Eine Umbenennung oder Übersetzung würde folglich dem gängigen Sprachgebrauch der Zielgruppe zuwiderlaufen und dadurch die praktische Verwendbarkeit der Ergebnisse verschlechtern.

## 5.1. Flexibilität

Der Schwerpunkt *Flexibilität* basiert auf den in Kapitel 2.4.2 identifizierten Anforderungen an die Erweiterbarkeit, Veränderbarkeit und Anpassbarkeit verteilter Krankenakten. Er beinhaltet Patterns, die helfen, ein Softwaresystem so zu gestalten, dass es leicht um zusätzliche Funktionen erweitert, flexibel mit anderen Systemen verbunden und an sich stetig verändernde Gegebenheiten angepasst werden kann. Der Schwerpunkt *Flexibilität* beschreibt hierzu ein durchgängiges Konzept, das ausgehend von einer problembezogenen Anforderungsanalyse die Konzeption eines bedarfsgerecht flexiblen Softwaresystems erlaubt.

### 5.1.1. Anforderungsanalyse

Die Analyse der Anforderungen ist bezüglich des Schwerpunkts Flexibilität in zwei Gruppen gegliedert. Die eine Gruppe enthält Patterns, die der Analyse der existierenden Rahmenbedingungen dienen, während die andere Gruppe Patterns beinhaltet, die sich mit der Identifikation und Dokumentation konkreter Anforderungen an die Flexibilität des zukünftigen Softwaresystems verteilte Krankenakte befassen.

Gruppen:

- Analyse flexibilitätsbeeinflussender Faktoren
- Ermittlung der Flexibilitätsanforderungen

Die Anforderungsanalyse beginnt mit der Anwendung des Patterns „Analyse der organisatorischen Verteilung im Verbund“ (Anhang S. 272) aus der Gruppe „Analyse flexibilitätsbeeinflussender Faktoren“. Das Pattern „Analyse der organisatorischen Verteilung im Verbund“ ist auch das Einstiegspattern für die Anforderungsanalyse der Schwerpunkte *Kommunikation* und *Zuverlässigkeit*. Es liefert nach seiner Anwendung einen Katalog der an der verteilten Krankenakte zu beteiligenden Einrichtungen, einschließlich ihrer Größe und der jeweils angebotenen Leistungen.

Ausgehend von dieser Basisinformation kommen die Patterns „Analyse der betroffenen Behandlungspfade“ (Anhang S. 267), „Untersuchung betroffener Subsysteme“ (Anhang S. 595) und „Analyse der Organisationsstruktur“ (Anhang S. 270) zum Einsatz, wobei Letzteres nicht direkt in die Ermittlung der Flexibilitätsanforderungen eingeht.

Bei der Analyse der betroffenen Behandlungspfade wird auf der Basis der ermittelten beteiligten Einrichtungen festgestellt, welche Behandlungspfade aufgrund der Rahmenbedingungen des Verbunds und des Leistungsumfangs der beteiligten Einrichtungen einrichtungsübergreifend abgebildet werden müssen. Die Untersuchung der betroffenen Subsysteme baut auf den bereits durch die Anwendung der beiden bisher verwendeten Analyse-Patterns gewonnenen Erkenntnissen auf. Auf Basis der Informationen über beteiligte Einrichtungen und Organisationseinheiten sowie der betroffenen Behandlungspfade wird ermittelt, welche der bestehenden Softwaresysteme der Einrichtungen vom Aufbau einer verteilten Krankenakte betroffen sind.

Die Ergebnisse aus der Untersuchung der betroffenen Subsysteme und der Analyse der betroffenen Behandlungspfade bilden, wie in Abbildung 36 dargestellt, die Ausgangsinformation für die Anwendung des Patterns *Veränderlichkeitsanalyse* (Anhang S. 599).

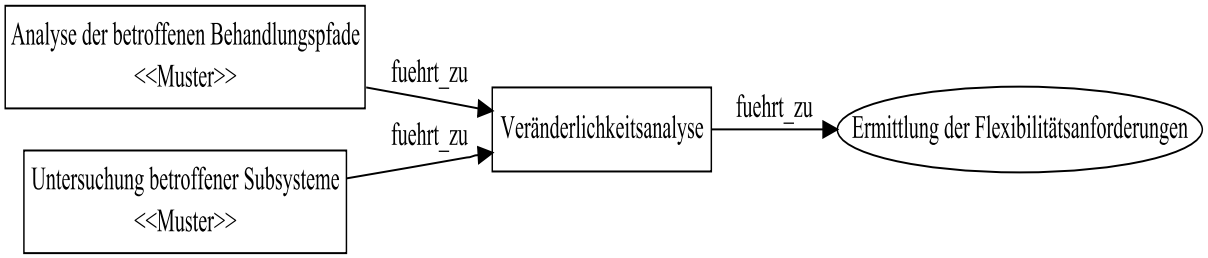



Abbildung 36: Abhängigkeitsgraph des Patterns Veränderlichkeitsanalyse

Für die Durchführung der Veränderlichkeitsanalyse stehen aufgrund der vorherigen Untersuchung betroffener Subsysteme Informationen über die einzelnen betroffenen Subsysteme zur Verfügung. Diese Informationen ermöglichen eine Bewertung der Wahrscheinlichkeit, mit der sich bestimmte Bestandteile des jeweils einzelnen Subsystems durch Einflussfaktoren, wie Software-Upgrades, Wechsel des Softwareprodukts oder Änderungen an verwendeten Schnittstellen, veränderlich sind. So ist z. B. eine Schnittstelle, die einen international verabschiedeten Standard implementiert, als weniger veränderlich zu betrachten als eine nur zur internen Verwendung vorgesehene proprietäre Schnittstelle eines einzelnen Softwareprodukts. Eine schon lange verwendete Software, deren technische Plattform bereits als veraltet zu bewerten ist, hat eine höhere Wahrscheinlichkeit, durch ein technisch aktuelleres Produkt ersetzt zu werden, als eine Software, die auf einer als zeitgemäß eingestuften Plattform basiert.

Behandlungspfade sind z. B. durch Veränderungen der Organisation eines Verbunds von Einrichtungen, den medizinischen Fortschritt oder durch gesetzliche Änderungen stetigen Änderungen unterworfen. Durch eine Synthese der ermittelten Veränderlichkeitspotenziale der Subsysteme und ihrer Bestandteile mit den Behandlungspfaden und den Veränderlichkeitspotenzialen ihrer Abschnitte ergibt sich ein Bild der Bereiche einer zukünftigen verteilten Krankenakte, in denen besonders hohe Flexibilität erforderlich ist.

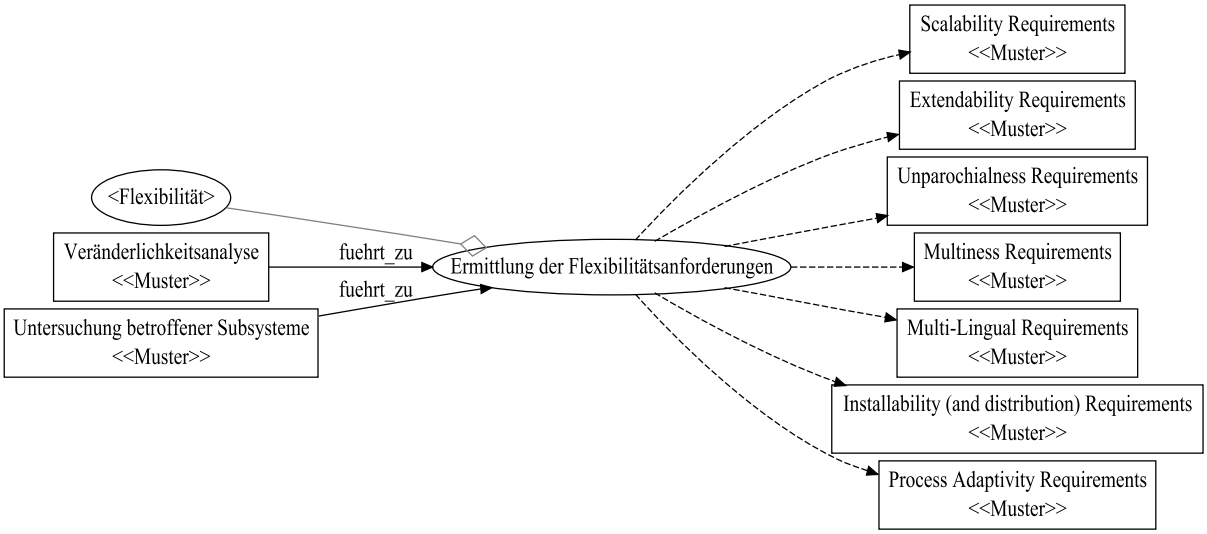



Abbildung 37: Direkte Abhängigkeiten der Gruppe "Ermittlung der Flexibilitätsanforderungen"

Die Ergebnisse der Veränderlichkeitsanalyse und der Untersuchung betroffener Subsysteme fließen direkt in die Ermittlung der Flexibilitätsanforderungen ein (siehe Abbildung 37). Die Ergebnisse der anderen Patterns der Gruppe „Analyse flexibilitätsbeeinflussender Faktoren“ werden entsprechend implizit berücksichtigt.

Die Gruppe „Ermittlung der Flexibilitätsanforderungen“ beinhaltet folgende Patterns:

- Extendability Requirements (Anhang S. 370)
- Installability (and distribution) Requirements (Anhang S. 407)
- Multi-Lingual Requirements (Anhang S. 454)
- Multiness Requirements (Anhang S. 455)
- Process Adaptivity Requirements (Anhang S. 476)
- Scalability Requirements (Anhang S. 541)
- Unparochialness Requirements (Anhang S. 593).

Diese verschiedenen Patterns dienen der Definition der Anforderungen, die die Flexibilität des Softwaresystems verteilte Krankenakte direkt beeinflussen. So beschreibt beispielsweise das Extendability-Requirements-Pattern, wie die gewünschte Erweiterungsfähigkeit gezielt als Anforderung dokumentiert werden kann. Das Pattern fordert, dass für jeden konkreten Aspekt, der erweiterbar gestaltet werden soll, eine entsprechende Anforderungs-dokumentation erfolgt. Das Installability-and-distribution-Requirements-Pattern befasst sich mit der Erfassung von Anforderungen bezüglich der Installierbarkeit, Verteilbarkeit und Lauf-fähigkeit der Software auf unterschiedlichen Plattformen oder unter verschiedenen Rahmen-bedingungen. So kann mithilfe der Patterns aus der Gruppe „Ermittlung der Flexibilitäts-anforderungen“ die Definition von Anforderungen zu verschiedenen die Flexibilität des zu planenden Systems beeinflussenden Aspekten vorgenommen werden. Die Umsetzung erfolgt, abhängig von den Inhalten der definierten Anforderungen, durch verschiedene Patterns der Ebene *Architektur*.

### 5.1.2. Architektur

Die Ebene *Architektur* im Schwerpunkt *Flexibilität* ist mit sieben Gruppen und neunzehn, teils mehrfach enthaltenen Patterns deutlich umfangreicher als die zugehörige Ebene *Anforderungsanalyse*.

Gruppen der Ebene *Architektur* des Schwerpunkts *Flexibilität*:

- Datenmodellabbildung
- Flexibilitätserhöhende Architekturmuster
- Grundlegende Architekturstile
- Prozessabbildung
- Suche im verteilten Dokumentensystem
- Verbindung zwischen Anwendungslogik und Daten
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur).

Die Gruppen beinhalten Patterns, die die Flexibilität des zu entwickelnden Softwaresystems bezüglich verschiedener Eigenschaften erhöhen. So kann beispielsweise ein durch den Anwender gestaltbares Datenmodell die Flexibilität der Anwendung hinsichtlich der Adaption neuer inhaltlicher Erfordernisse verbessern. Eine wohlstrukturierte Softwarearchitektur erleichtert die Veränderung des Softwaresystems durch Anpassung des Quellcodes. Die Zerlegung komplexer Probleme in leichter zu lösende Teilprobleme führt zu einer Vereinfachung bei der Umsetzung der Software. Durch die sinnvoll gegliederten Bestandteile gestaltet sich aber auch die Wartung und die Anpassung einfacher. Damit nähern sich die verschiedenen Gruppen dieses Schwerpunkts und dieser Ebene aus verschiedenen Richtungen der flexiblen Gestaltung der Softwarearchitektur an. Sie können, miteinander kombiniert, entscheidend dazu beitragen problemadäquate Flexibilität zu erreichen.

Im Softwareengineering haben sich im Laufe der Zeit verschiedene Grundprinzipien zur Gestaltung der Architektur von Softwaresystemen durchgesetzt (vgl. Prinzipien des Architekturentwurfs in Ludewig & Lichter 2010:S. 410–442). Die Gruppe „Grundlegende Architekturstile“ ist eine Sammlung solcher in Patternform beschriebener Grundprinzipien.

Patterns der Gruppe „Grundlegende Architekturstile“:

- Central Message Bus (Anhang S. 307)
- Komponentenbasierte Softwarearchitektur (Anhang S. 419)
- Microkernel (Anhang S. 446)
- Model View Controller (Anhang S. 450)
- Pipes and Filters (Anhang S. 471)
- Schichtenarchitektur (Anhang S. 543).

Die Abbildung 38 zeigt die Gruppe „Grundlegende Architekturstile“ mit ihren Abhängigkeiten zu Patterns der Gruppe „Ermittlung der Flexibilitätsanforderungen“ und den Beziehungen zwischen den dargestellten Patterns.

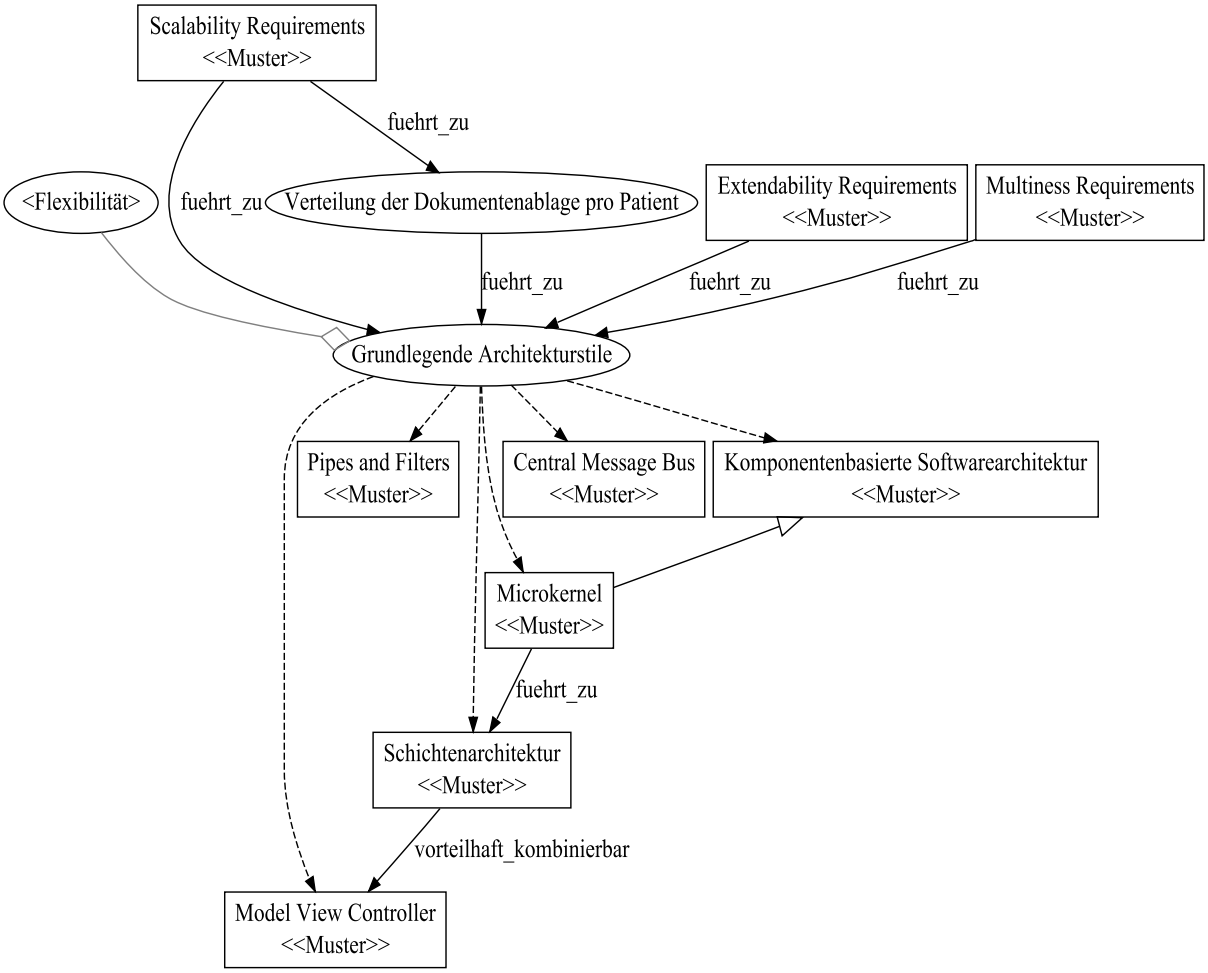



Abbildung 38: Gruppe „Grundlegende Architekturstile“ mit internen Beziehungen

Die Patterns zu den Architekturstilen eignen sich für verschiedene Typen von Softwaresystemen oder Systembestandteilen. Sie sind gut kombinierbar und können sowohl auf der Ebene des Gesamtsystems, eines Subsystems oder eines noch kleineren Systembestandteils eingesetzt werden.

Die Kombinierbarkeit der grundlegenden Architekturprinzipien auf unterschiedlichen Granularitätsebenen lässt sich anhand eines Beispiels erläutern. Das Gesamtsystem des Beispiels ist zur Kommunikation mit einem Central Message Bus verbunden. Es verwendet einen zentralen Message-Server zur Kommunikation. Diese Form der Systemarchitektur ermöglicht eine besonders lose Kopplung zwischen den beteiligten Subsystemen, da sie nicht direkt miteinander, sondern indirekt über den Message-Server kommunizieren. Die kommunizierenden Subsysteme sind selbst in eine Schichtenarchitektur gegliedert. Deren Benutzeroberfläche ist in Form des Model-View-Controller-Patterns strukturiert. Für Abfragen in dem System gibt es eine spezielle Abfragesprache, deren Interpreter entsprechend des Pipes-and-Filters-Patterns aufgebaut ist.

Dieses Beispiel zeigt, wie durch Kombination von Patterns der vorliegenden Gruppe auf verschiedenen Ebenen unterschiedliche Bestandteile eines Softwaresystems jeweils eine geeignete Grundstruktur für ihre Softwarearchitektur erhalten können. Ein beabsichtigtes Nebenprodukt einer übersichtlich strukturierten Softwarearchitektur ist ihre verbesserte Wartbarkeit und Veränderbarkeit.

Neben der Gruppe der grundlegenden Prinzipien gibt es eine Gruppe der Architekturmuster, deren Patterns nahezu ausschließlich der Verbesserung der Flexibilität von Softwaresystemen dienen. Diese Patterns werden in der Gruppe der flexibilitätserhöhenden Architekturmuster zusammengefasst.

Patterns der Gruppe „Flexibilitätserhöhende Architekturmuster“:

- Adapter Registry (Anhang S. 254)
- Central Message Bus (Anhang S. 307)
- Microkernel (Anhang S. 446)
- Multi-Channel Access Provider (Anhang S. 452).

Das Adapter-Registry-Pattern dient der Integration einer Vielzahl ähnlicher Systeme und unterstützt so die Schaffung einer übergeordneten Anwendungslogik, die die gleichzeitige Bedienung der Gesamtheit der angebundenen Systeme oder einer Teilmenge daraus erlaubt.

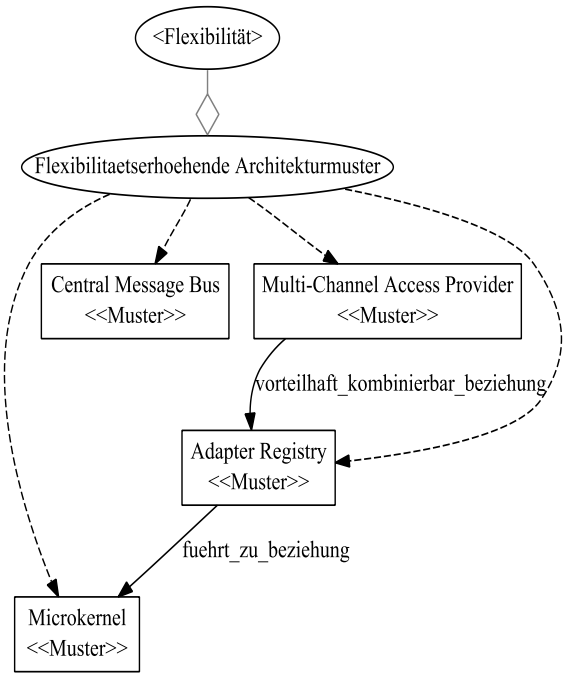



Abbildung 39: Gruppe „Flexibilitätserhöhende Architekturmuster“ mit internen Beziehungen

Die Abbildung 39 zeigt, dass sich das Multi-Channel-Access-Provider-Pattern gut mit dem Adapter-Registry-Pattern kombinieren lässt. Ein praktisches Beispiel für diese Kombination ist die Softwarearchitektur des Einweiserportals des Universitätsklinikums Regensburg (siehe Kapitel 6.2 und (Wiedermann u. a. 2008; Wiedermann u. a. 2009)).

Eine weitere Aufgabe der Softwarearchitektur ist es, komplexe Aufgaben und Konstrukte in beherrschbare Teilmengen zu zerlegen. Diese Zerlegung führt, wie bereits auf S. 137 erwähnt, zu einer verbesserten Veränderbarkeit und Anpassbarkeit der Software und fördert so die Flexibilität des resultierenden Softwaresystems. Die zugehörige Gruppe beinhaltet folgende Patterns.

Patterns der Gruppe „Zerlegung komplexer Aufgaben und Konstrukte“:

- Business Process Layer (Anhang S. 304)
- Database Access Layer (Anhang S. 338)
- Komponentenbasierte Softwarearchitektur (Anhang S. 419)
- Microkernel (Anhang S. 446)
- Model View Controller (Anhang S. 450)
- Pipes and Filters (Anhang S. 471)
- Schichtenarchitektur (Anhang S. 543)
- Service Layer (Anhang S. 554).

Die Abbildung 40 zeigt die Gruppe „Zerlegung komplexer Aufgaben und Konstrukte“ der Ebene *Architektur* einschließlich der Beziehungen zwischen den enthaltenen Patterns.

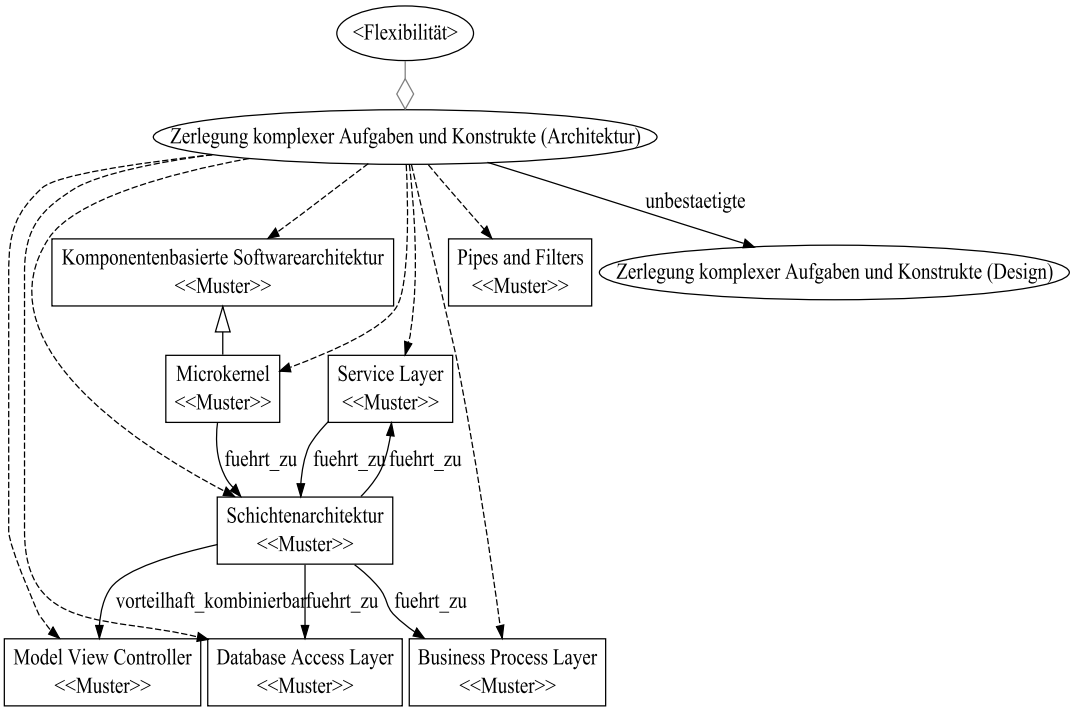



Abbildung 40: Zerlegung komplexer Aufgaben und Konstrukte mit internen Abhängigkeiten

Eine Zerlegung komplexer Probleme und Aufgaben kann durch verschiedene Mechanismen erfolgen. Ein besonders häufig verwendeter Ansatz ist die Zerlegung in Schichten entsprechend dem Schichtenarchitektur-Pattern. Hierzu bietet die Gruppe „Zerlegung komplexer Aufgaben und Konstrukte“ neben dem Basispattern Schichtenarchitektur auch verschiedene Patterns zur Bildung konkreter Schichten wie z. B. das Service-Layer-, Database-Access-Layer- und Business-Process-Layer-Pattern. Das Business-Process-Layer-Pattern ist ein besonders geeigneter Kandidat, um die Auswirkung der logischen Zerlegung auf die Flexibilität des resultierenden Softwaresystems zu unterstreichen. Durch die Abbildung der prozessbezogenen Aspekte in einer eigenen Schicht wird es ermöglicht, Änderungen in der Flusslogik von Prozessen an einer zentralen Stelle durchzuführen. Diese Eigenschaft führt dann zu besonderer Flexibilität, wenn in dieser Schicht die Definition der Ablauflogik des Prozesses durch zusätzliche Mechanismen konfigurierbar gestaltet wird.

Die Database Access Layer stellt eine spezielle Schicht zur Trennung zwischen Schichten, die Daten nutzen, und der Schicht der Datenhaltung bzw. Datenspeicherung dar. Sie kann beispielsweise verwendet werden, um die Schichten der Anwendungslogik unabhängig von der konkreten Form der Datenspeicherung zu machen. Für den Aspekt der Trennung zwischen Anwendungslogik und Daten existiert im Schwerpunkt *Flexibilität* eine eigene Gruppe.

Patterns der Gruppe „Verbindung zwischen Anwendungslogik und Daten“:

- Data Access Object (Anhang S. 333)
- Database Access Layer (Anhang S. 338).

Die Gruppe enthält zwei Patterns, das Data-Access-Object-Pattern, ein Pattern der Ebene Design, und das Database-Access-Layer-Pattern als Pattern der Ebene *Architektur*. Die Kombination von Patterns unterschiedlicher Ebenen in einer Gruppe ist ungewöhnlich, in diesem Fall aber sinnvoll, da so zwei zusammengehörige Patterns als eine Einheit dargestellt werden können. Die Schicht der Database Access Layer kann auf der Design-Ebene durch die Anwendung des Data-Access-Object-Patterns umgesetzt werden.

Zusätzlich zur Verbindung zwischen Anwendungslogik und Daten wird die Flexibilität des Softwaresystems, insbesondere einer verteilten Krankenakte, auch durch die Art der technischen Abbildung des Datenmodells beeinflusst.

Patterns der Gruppe „Datenmodellabbildung“:

- Domain Model (Anhang S. 351)
- Entity-Attribute-Value (Anhang S. 360)
- Object Relational Mapping (Anhang S. 460).

Bei den Patterns der Gruppe „Datenmodellabbildung“ handelt es sich um relativ feingranulare, detailliert beschreibbare Architekturpatterns, die sich im Grenzbereich zu den Designpatterns befinden. Sie beschreiben bekannte Verfahrensweisen zur technischen Abbildung von Datenmodellen der Zieldomäne auf Datenbanken. Die vorgestellten Patterns beschränken sich hierbei auf die Abbildung des Modells auf relationalen Datenbanken. Zur Erreichung einer umfangreicheren fachlichen Vollständigkeit kann die Gruppe um beliebige weitere Mechanismen zur Abbildung von Datenmodellen z. B. in NoSQL-Datenbanken oder XML-Dateien erweitert werden.

Als besonders flexibles, dafür aber bezüglich Zuverlässigkeit und Performance nachteiliges Muster kann das enthaltene Entity-Attribute-Value-Pattern herausgestellt werden. Einerseits ermöglicht es prinzipiell, bei jedem erfassten Datensatz oder medizinischen Dokument die Anzahl und die Art der Spalten zu variieren, ohne dass das Schema bereits erfasster Sätze oder Dokumente dadurch in Mitleidenschaft gezogen wird. Es können aber auch Zeilen entstehen, die durch die gegebenen Programme nicht richtig verarbeitbar sind. Zudem unterstützt das Pattern in seiner Grundform keine typisierte Speicherung der Daten in den Feldern und erfordert eine große Zahl von Join-Operationen für das Auslesen eines einzelnen

Datensatzes. Diese Eigenschaften führen – im Vergleich zur klassisch relationalen Abbildungsform des Domain Model – zu einer schlechteren Laufzeiten für Abfragen und Auswertungen (Chen u. a. 2000).

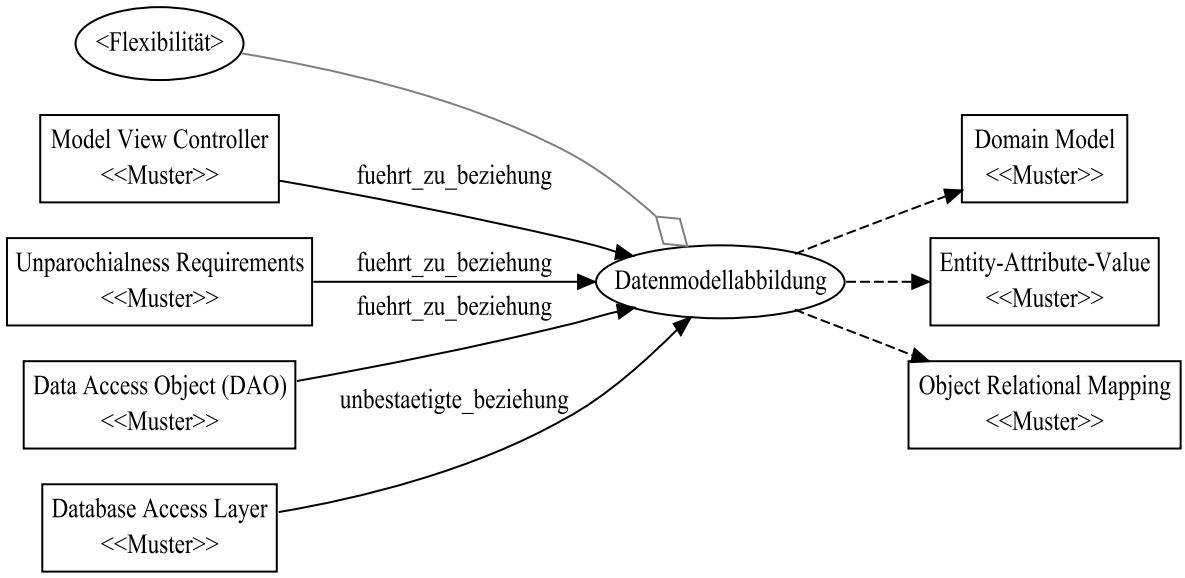



Abbildung 41: Gruppe „Datenmodellabbildung“ ohne interne Beziehungen

Der in Abbildung 41 dargestellte gerichtete Graph zeigt die Gruppe „Datenmodellabbildung“ einschließlich ihrer eingehenden Beziehungen und enthaltenen Patterns. Er zeigt, dass der Einsatz verschiedener anderer Patterns die Verwendung eines Patterns der Gruppe „Datenmodellabbildung“ erfordern oder, wie das Pattern „Unparochialness Requirements“, dessen Auswahl beeinflussen kann. Das Unparochialness-Requirements-Pattern dient der Spezifikation von Anforderungen an ein System, sich flexibel den Bedingungen unterschiedlicher organisatorischer oder fachlicher Umgebungen anzupassen. Eine veränderte Einsatzumgebung führt häufig zu Veränderungen in den zu erfassenden Daten, da durch abweichende Verfahrensweisen oder Nachvollziehbarkeitsanforderungen möglicherweise zusätzliche Attribute erfasst werden müssen, während andere ihre Notwendigkeit verlieren. Ein flexibel anpassbares Datenmodell verbessert folglich auch die Fähigkeit eines Systems, flexibel in verschiedenen Umgebungen eingesetzt werden zu können.

Verändert sich die organisatorische oder fachliche Umgebung, so besteht eine erhöhte Wahrscheinlichkeit, dass sich zugleich auch eine Notwendigkeit für die Anpassung von im System abgebildeten Prozessen ergibt. Die zugehörigen Patterns sind in der Gruppe „Prozessabbildung“ zusammengefasst.

Patterns der Gruppe „Prozessabbildung“:

- Business Process Layer (Anhang S. 304)
- Distributed Business Process (Anhang S. 346).

Im Rahmen der verteilten Krankenakten zugrunde liegenden integrierten Versorgung finden Behandlungsprozesse häufig einrichtungsübergreifend statt. Um diesem Umstand Rechnung

zu tragen, enthält die Gruppe das Distributed-Business-Process-Pattern, das einen Architekturansatz zur aus Benutzersicht nahtlosen Integration von Subsystemen in einen gemeinsamen Prozess beschreibt.

Die Flexibilität, mit der eine verteilte Krankenakte um zusätzliche Arten von Dokumenten oder um zusätzliche Einrichtungen erweitert werden kann, ist auch direkt vom verwendeten Suchmechanismus und dessen zugrunde liegender Kommunikationsarchitektur abhängig.

Patterns der Gruppe „Suche im verteilten Dokumentensystem“:

- Flooding based Search (Anhang S. 387)
- Zentraler Suchdienst (Anhang S. 612).

Mit den beiden Patterns „Flooding based Search“ und „Zentraler Suchdienst“ enthält die Gruppe zwei Extreme, die sich durch unterschiedliche Eigenschaften klar voneinander abgrenzen und innerhalb unterschiedlicher Einsatzbedingungen ihre Vorteile besitzen. Der Ansatz „Flooding based Search“ stammt aus dem Bereich der Peer-to-Peer-Netzwerke, in denen kein Knoten absolutes Wissen über alle aktuell im System befindlichen Knoten besitzt. Suchanfragen werden vielmehr von Knoten zu Knoten weitergereicht, indem jeder Knoten die Anfrage an alle ihm bekannten Knoten übergibt. Der Vorteil dieses Verfahrens liegt darin, dass zur Durchführung einer Suche kein Verzeichnis über alle verfügbaren Systemknoten und deren Inhalte existieren muss. Die zwei wesentlichen Nachteile liegen zum einen in der relativ niedrigen Performance der Suchen, ausgelöst durch die Notwendigkeit die Suchanfrage auch auf einer großen Zahl von Knoten auszuführen, die keine für die Suche relevanten Inhalte besitzen. Zum anderen besteht in Peer-to-Peer-Netzwerken mit nicht ausreichendem Vernetzungsgrad zwischen den Knoten das Risiko, dass Mengen von Knoten mit relevanten Inhalten nicht gefunden werden, da es keine verfügbare Route zwischen dem suchenden Knoten und einer abgegrenzten Menge relevanter Knoten gibt.

Bessere Zuverlässigkeit bei gleichzeitig besserer Performance sind die wesentlichen Vorteile, die aus der Anwendung des Patterns „Zentraler Suchdienst“ resultieren. Seine Verwendung setzt allerdings den Aufbau eines Index über alle beteiligten Systemknoten und deren Inhalte voraus.

### 5.1.3. Design

Architekturpatterns geben eine grobe Struktur für die Lösung von Problemen und die Erlangung bestimmter Eigenschaften vor. Um der Erreichung der gewünschten Eigenschaften einen weiteren Schritt näher zu kommen, ist es notwendig, diese Strukturen durch die Anwendung feingranularerer Lösungsansätze zu füllen. Dazu dienen Design-Patterns.

Für den Schwerpunkt *Flexibilität* werden in diesem Kapitel neun Gruppen, mit insgesamt 31 Design-Patterns vorgestellt.

Gruppen des Schwerpunkts *Flexibilität* in der Ebene *Design*:

- Datentransformation
- Instanziierung und Kontrolle der Instanzen
- Kopplung und Entkopplung
- Statische Veränderung der Schnittstelle
- Verwaltung des Prozesszustands
- Zerlegung komplexer Aufgaben und Konstrukte
- Sonstige flexibilitätserhöhende Design-Patterns
- Handle Body Patterns.

Die Integration von Softwaresystemen setzt Schnittstellen zwischen bisher eigenständig betriebenen Systemen voraus. Die Daten- und Objektmodelle der betreffenden Systeme und ihre bereits existierenden Schnittstellen müssen nicht zwangsläufig direkt zueinander kompatibel sein. Aber auch wenn eine solche Kompatibilität nicht vorliegt, muss die Kommunikation ermöglicht werden. Um Kompatibilitätsprobleme zwischen den Datenstrukturen von Nachrichten zu beheben, kann zum Mittel der Datentransformation gegriffen werden. Die Datentransformation erhöht somit die Flexibilität bei der Kommunikation zwischen heterogenen Systemen.

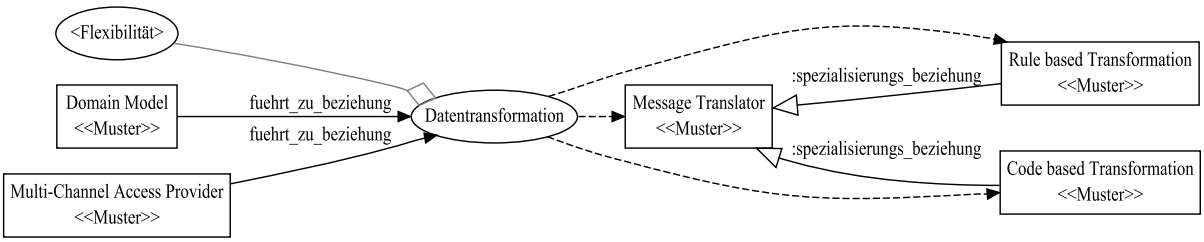



Abbildung 42: Gruppe „Datentransformation“ mit Beziehungen zwischen direkt von der Gruppe abhängigen Objekten

Die Gruppe Datentransformation beinhaltet drei Patterns. Das Message-Translator-Pattern ist das Basispattern der Gruppe. Die beiden anderen Patterns sind Spezialformen des Message-Translator-Patterns (siehe Abbildung 42).

Patterns der Gruppe „Datentransformation“:

- Code based Transformation (Anhang S. 319)
- Message Translator (Anhang S. 439)
- Rule based Transformation (Anhang S. 540).

In objektorientiert entwickelten Softwaresystemen enthalten Objekte als konkrete Exemplare von Klassen (Jobst 2001:S. 75) sowohl Daten als auch Operationen zur Manipulation der enthaltenen Daten. Jedes dieser Objekte nimmt entsprechend seiner Datenstrukturen Platz im Hauptspeicher des Rechners ein, auf dem das Softwaresystem ausgeführt wird. Die Erzeugung von Exemplaren (Objekten) von Klassen nennt man auch Instanziierung (Jobst 2001:S. 79).

In derzeit verbreiteten Programmiersprachen wie Java oder C# gibt es verschiedene Mechanismen zur Instanziierung, die sich durch ihre Flexibilität aber auch ihren nötigen Codeumfang unterscheiden. Einerseits existiert der klassische, statische und für die meisten Anwendungsfälle geeignete Ansatz der Instanziierung durch ein Schlüsselwort (z. B. new) gefolgt von der zu instanziierenden Klasse und deren Konstruktorparametern (Jobst 2001:S. 79). Dieser Ansatz ermöglicht die Festlegung der zu instanziierenden Klasse ausschließlich zum Zeitpunkt der Codierung des Systems. Andererseits existieren auch Methoden zur Instanziierung von Klassen, deren Namen durch Variable festgelegt werden. In Java ist das durch die Kombination der Methoden forName[15] und newInstance[16] der Klasse Class möglich. Diese Methoden erhöhen die Flexibilität einer Anwendung, da sie die Entscheidung, welche konkrete Implementierung z. B. einer Schnittstelle verwendet werden soll, vom Zeitpunkt der Codierung des Systems in dessen Laufzeit verlegen. Gleichzeitig machen sie allerdings den Code komplexer und sind schwerer zuverlässig zu beherrschen. Aus diesem Grund sind Patterns zur Vereinfachung der Entstehung, der Reglementierung der Anzahl und der planmäßigen Vernichtung von Objekten notwendig.

Patterns der Gruppe „Instanziierung und Kontrolle der Instanzen“:

- Abstract Factory (Anhang S. 244)
- Factory Method (Anhang S. 375)
- Disposal Method (Anhang S. 345)
- Prototype (Anhang S. 484)
- Singleton (Anhang S. 571).

Die Muster zur Generierung von Objekten, Abstract Factory und Factory Method, eignen sich z. B. für die Umsetzung von Plug-in-Mechanismen. Die Aufgaben einer Abstract Factory sind so gestaltet, dass es meist nicht sinnvoll ist, mehrere Instanzen der Abstract Factory zu generieren. Die Abbildung 43 zeigt, dass dieses Problem durch die Kombination des Patterns „Abstract Factory“ mit dem ebenfalls in der Gruppe enthaltenen Singleton-Pattern gelöst werden kann.

[15] http://download.oracle.com/javase/7/docs/api/java/lang/Class.html#forName(java.lang.String)
[16] http://download.oracle.com/javase/7/docs/api/java/lang/Class.html#newInstance()

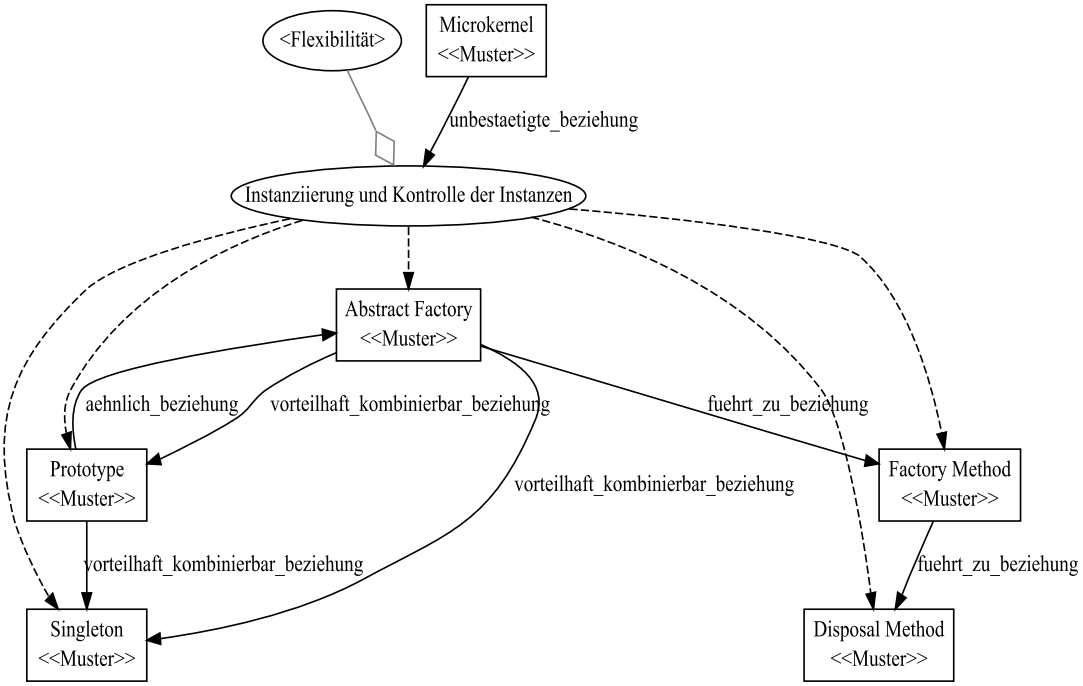



Abbildung 43: Gruppe „Instanziierung und Kontrolle der Instanzen“ mit internen Beziehungen

Durch Patterns zur dynamischen, von Variablenwerten bestimmten Instanziierung von Klassen kann zur Entkopplung von Komponenten beigetragen werden. Unter Entkopplung ist zu verstehen, dass der Code einer Komponente A von dem einer Komponente B unabhängiger gemacht wird. Diese Unabhängigkeit ist nicht allein von der Instanziierung abhängig; auch viele andere Faktoren beeinflussen die Abhängigkeiten zwischen zwei Komponenten. Die Gruppe „Kopplung und Entkopplung“ liefert eine Menge von Patterns, die bezüglich unterschiedlicher Aspekte, zur Vergrößerung der Unabhängigkeit zwischen Softwarebausteinen beitragen können.

Pattern der Gruppe „Kopplung und Entkopplung“:

- Adapter (Anhang S. 252)
- Bridge (Anhang S. 299)
- Broker (Anhang S. 300)
- Client-Dispatcher-Server (Anhang S. 317)
- Command (Anhang S. 321)
- Component Configurator (Anhang S. 325)
- Facade (Anhang S. 372)
- Lookup (Anhang S. 429)
- Proxy (Anhang S. 485)
- Registry (Anhang S. 505).

Im praktischen Einsatz von Softwaresystemen wie verteilten Krankenakten ist es häufig notwendig, dass Softwarekomponenten, die nicht für den gemeinsamen Einsatz konzipiert, also nicht zueinander kompatibel sind, gemeinsam verwendet werden. In diesem Fall liegt die zentrale Aufgabe in der Überwindung nicht kompatibler Schnittstellen. Auch dazu können die Patterns der Gruppe „Kopplung und Entkopplung“ genutzt werden.

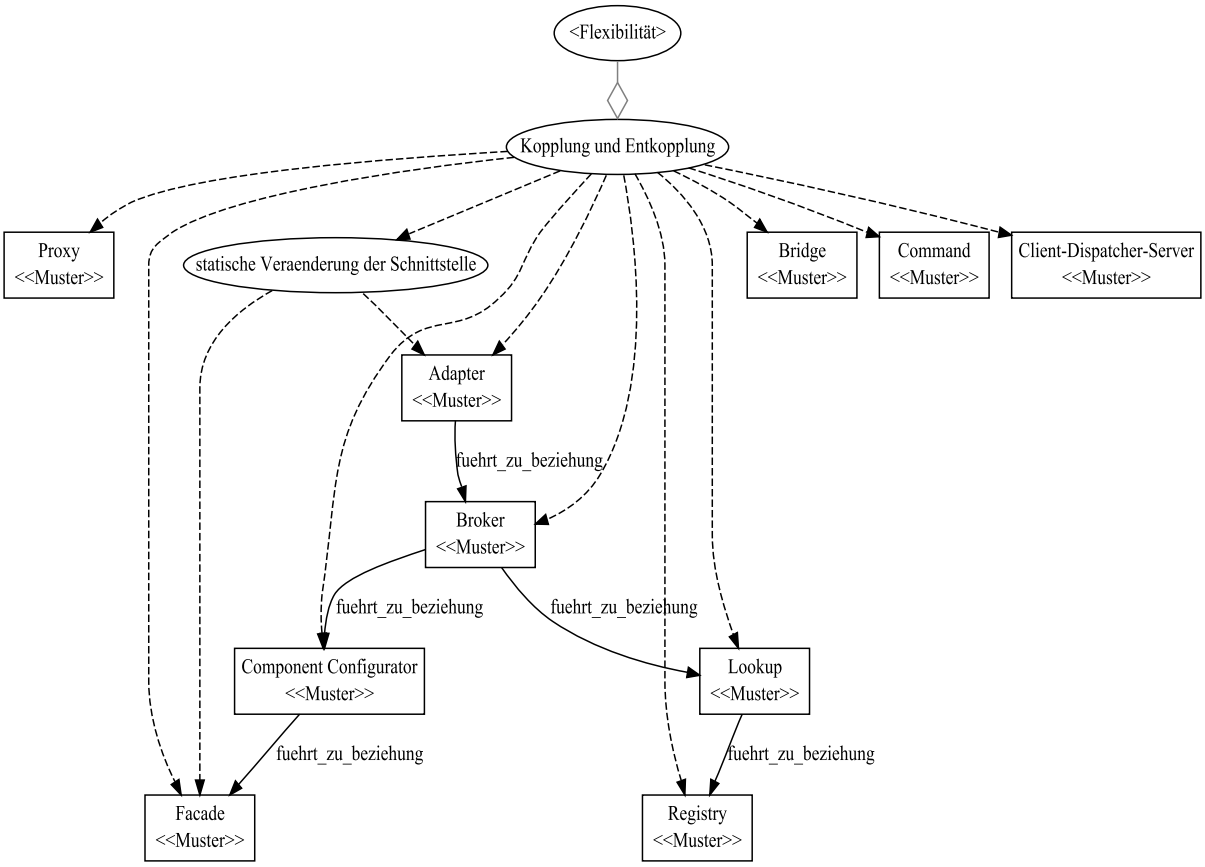



Abbildung 44: Gruppe „Kopplung und Entkopplung“ ohne eingehende Beziehungen

Die Gruppe besitzt den Namen *Kopplung und Entkopplung*, weil die enthaltenen Mechanismen bei nicht aufeinander abgestimmten Komponenten dazu dienen können, die Kommunikation zwischen diesen Komponenten zu ermöglichen. Hierzu ist eine Adaption der jeweils vom Kommunikationspartner erwarteten Schnittstelle, die Vermittlung der Nachrichten an die richtige Schnittstelle, z. B. über einen Stellvertreter, und vieles mehr nötig. Zwei oder mehrere vorher inkompatible Komponenten werden durch diese Mechanismen kombinierbar gemacht oder aneinander gekoppelt. Bei der Entwicklung eines vollständig neuen Softwaresystems können die gleichen Mechanismen verwendet werden, um die Bestandteile des Systems stärker voneinander unabhängig zu machen. Das vermeidet wiederum unnötige Abhängigkeiten, die sich bei Änderungen negativ auswirken können.

Ein Teilbereich der Gruppe „Kopplung und Entkopplung“ befasst sich mit der statischen Veränderung der Schnittstellen von Klassen oder Komponenten (siehe Abbildung 44). Die dafür vorgesehenen Patterns sind deshalb zusätzlich in der Gruppe „Statische Veränderung der Schnittstelle“ zusammengefasst.

Patterns der Gruppe „Statische Veränderung der Schnittstelle“:

- Adapter (Anhang S. 252)
- Decorator (Anhang S. 341)
- Facade (Anhang S. 372).

Das Adapter-Pattern beschreibt einen etablierten Weg, um das Verhalten einer anderen, bekannten Schnittstelle anzunehmen. Daneben enthält die Gruppe auch ein Pattern zum Verbergen von Komplexität und zum Erstellen von problembezogenen Schnittstellen (Fassade). Ein anderes enthaltenes Pattern (Decorator) kann verwendet werden, um bereits existierenden Klassen oder Komponenten zusätzliche Funktionalität hinzuzufügen. Die Möglichkeit der Veränderung von Schnittstellen erhöht die Flexibilität eines Softwaresystems durch eine Verbesserung der Kombinierbarkeit verschiedener Komponenten.

Ein weiterer Aspekt, der bei verteilten Krankenakten einen hohen Flexibilitätsbedarf aufweist, ist die systemseitige Abbildung von Prozessen. Die in Kapitel 5.1.2 beschriebenen Architekturpatterns zur Integration prozessspezifischer Bestandteile in die Systemarchitektur erfordern, neben weiteren Mechanismen, für ihre technische Umsetzung Wege, um den Zustand von Prozessinstanzen zu verwalten und zu ermitteln. Dafür gibt es verschiedene Lösungsansätze, die sich vor allem durch den unterschiedlichen Aufwand bei der Umsetzung und eine unterschiedliche Flexibilität bei Veränderungen unterscheiden.

Patterns der Gruppe „Verwaltung des Prozesszustands“:

- Persistent State Pattern (Anhang S. 470)
- Process by Object State (Anhang S. 478)
- Process State Object (Anhang S. 479).

Ein häufig verwendeter Ansatz zur Ermittlung des Prozesszustands ist die Ermittlung aus den Attributwerten beteiligter Objekte (Process by Object State). Dieser Ansatz gestaltet sich in seiner praktischen Umsetzung sehr vielfältig und reicht von komplexen Berechnungen auf der Basis fachlich bedingter Attribute bis hin zur Führung eines speziellen Attributs zur Zwischenspeicherung des jeweils aktuellen Prozesszustands.

Ein anderer Ansatz folgt dem Process-State-Object-Pattern und sieht zur Speicherung des Prozesszustands ein separates Objekt pro Prozessinstanz vor. Während beim Process-by-Object-State-Pattern der Prozesszustand inhärenter Bestandteil des Domänenmodells ist, kann das Process State Object auch in einer separaten Schicht, z. B. der Business Process Layer, unabhängig vom eigentlichen Domänenmodell verwaltet werden. Dadurch erhöht sich die Flexibilität des Systems, da gleiche Objekte prinzipiell auch gleichzeitig in verschiedenen Prozessen verwendet werden können und die Verwendung der Klassen in den Prozessen zur Entwicklungszeit des Domänenmodells nicht bekannt sein muss. Das Persistent-State-Pattern ist eine allgemeinere Form des Process-State-Object-Patterns.

Im Rahmen medizinischer Versorgungsprozesse entstehen Dokumente (vgl. z. B. Haas 2006:S. 185–190). In verteilten Krankenakten sind diese Dokumente per Definition verteilt auf verschiedene Systemknoten gespeichert. Die Suche im verteilten Dokumentensystem erfordert bei der Anwendung des Patterns „Zentraler Suchdienst“ den Aufbau eines zentralen Index über durchsuchbare Objekte und ihre Ablageorte. Dazu enthält die Gruppe „Verwaltung verteilter Dokumente“ das Document-Registry-Pattern (siehe Anhang S. 349).

Die Entwicklung verteilter Krankenakten ist eine komplexe Aufgabe. Auch die Subsysteme und Komponenten eines solchen Softwaresystems sind in sich noch sehr komplex. Große und komplexe Systembestandteile sind allerdings schwer durchschaubar und bergen im Falle einer Anpassung oder Veränderung das Risiko, vom Entwickler nicht ausreichend verstanden zu sein. Fehlendes Verstehen kann verhindern, dass eine Änderung im Sinne des ursprünglichen Entwurfs fehlerfrei umgesetzt wird. Aus diesem Grund ist es vorteilhaft, komplexe Aufgaben und Konstrukte innerhalb von Softwarekomponenten in bekannte Lösungsmuster zu zerlegen.

Patterns der Gruppe „Zerlegung komplexer Aufgaben und Konstrukte (Design)“:

- Builder (Anhang S. 302)
- Chain of Responsibility (Anhang S. 309)
- Composite (Anhang S. 327)
- Flyweight (Anhang S. 389)
- Iterator (Anhang S. 417)
- Strategy (Anhang S. 582)
- Visitor (Anhang S. 601).

Neben den bisher genannten Design-Patterns sind während der Analyse des Schwerpunkts *Flexibilität* noch weitere Patterns identifiziert worden, die sich nicht direkt den vorgestellten Gruppen zuordnen lassen, die aber direkt zur Verbesserung der Flexibilität von Softwaresystemen beitragen. Sie sind aus diesem Grund in einer offenen Gruppe mit der Bezeichung „Sonstige flexibilitätserhöhende Design-Patterns“ zusammengefasst.

Sonstige flexibilitätserhöhende Design-Patterns:

- Interpreter (Anhang S. 413)
- Meditator (Anhang S. 436)
- Memento (Anhang S. 438)
- Observer (Anhang S. 461)
- State (Anhang S. 578)
- Template Method (Anhang S. 587)
- Visitor (Anhang S. 601).

Eine besondere Sammlung stellt die Gruppe der Handle-Body-Patterns (Portland Pattern Repository 2009) dar. Diese Patterns sind in ihr nicht aufgrund ihres gemeinsamen Aufgabenbereichs, sondern aufgrund der Ähnlichkeit der in ihnen enthaltenen technischen Lösung gruppiert. Die Verwandtschaft der Handle-Body-Patterns besteht darin, dass in ihnen die Implementierung, genannt *Body*, vom Mittel des Zugriffs, genannt *Handle*, getrennt ist. Ein Beispiel für ein Handle-Body-Pattern mit hohem Bekanntheitsgrad ist das Remote-Proxy-Pattern. In ihm ist das die Implementierung beinhaltende Objekt vom zugriffsgewährenden Objekt getrennt realisiert. Beide sind ausschließlich über Netzwerkkommunikation miteinander verbunden.

Patterns der Gruppe „Handle Body Patterns“:

- Adapter (Anhang S. 252)
- Bridge (Anhang S. 299)
- Decorator (Anhang S. 341)
- Facet (Anhang S. 374)
- Half Object Plus Protocol (Anhang S. 398)
- Remote Proxy (Anhang S. 512).

Die Gruppe der Handle-Body-Patterns ist dem Schwerpunkt *Flexibilität* zugeordnet, da sie die Implementierung verpacken und in Form des Handles eine von der Implementierung stärker unabhängige Schnittstelle zur Verfügung stellen. Je nach verwendetem Pattern kann dadurch entweder die Gestaltung des Kommunikationswegs oder die Gestaltung der Schnittstelle flexibler werden.

## 5.2. Kommunikation

Die Gestaltung funktionierender Kommunikationsmechanismen ist die zentrale Aufgabe bei der Konzeption und Umsetzung einer verteilten Krankenakte. Als verteiltes System basiert ihre gesamte aus Sicht der Benutzer relevante Funktionalität auf der zuverlässigen entfernten Kommunikation zwischen ihren Systembestandteilen. Auch im bereits dargestellten Schwerpunkt *Flexibilität* betreffen viele berücksichtigte Aspekte die Kommunikation zwischen Bestandteilen des Softwaresystems verteilte Krankenakte. Sie befassen sich insbesondere mit der Erreichung einer möglichst flexiblen Verwendbarkeit verschiedener Kommunikationsmechanismen. Verschiedene Eigenschaften der Kommunikation betreffen auch die Schwerpunkte Sicherheit und Zuverlässigkeit.

Der in diesem Kapitel vorgestellte Schwerpunkt *Kommunikation* befasst sich primär mit Mustern zur Gestaltung und Erreichung von Kommunikationsfähigkeit in verteilten Systemen. Die Sammlung der Gruppen und Patterns im Schwerpunkt *Kommunikation* bildet, ebenso wie der bereits beschriebene Schwerpunkt *Flexibilität*, in der Abfolge der Ebenen betrachtet, ein geführtes Vorgehen zur Gestaltung der Kommunikationsarchitektur und der konkreten Kommunikationsmechanismen einer verteilten Krankenakte.

### 5.2.1. Anforderungsanalyse

Die Ergebnisse einer umfassenden Anforderungsanalyse sind die Basis für die problemorientierte Gestaltung der Kommunikation innerhalb einer verteilten Krankenakte. Die Anforderungsanalyse umfasst sowohl eine eingehende Analyse der Umgebung der zu entwickelnden verteilten Krankenakte als auch die Identifikation und Spezifikation konkreter Anforderungen. Deshalb sind die für die Anforderungsanalyse relevanten Patterns in die Gruppen „Analyse des Kommunikationsbedarfs“ und „Kommunikationsanforderungen“ aufgeteilt.

Gruppen der Anforderungsanalyse:

- Analyse des Kommunikationsbedarfs
- Kommunikationsanforderungen.

Die Anforderungsanalyse beginnt mit der Analyse der Rahmenbedingungen der verteilten Krankenakte, in der die Patterns der Gruppe „Analyse des Kommunikationsbedarfs“ angewendet werden. Im ersten Schritt wird die Analyse der organisatorischen Verteilung im Verbund durchgeführt. Durch die Anwendung dieses Patterns werden alle an der verteilten Krankenakte zu beteiligenden Einrichtungen und Organisationseinheiten, einschließlich ihrer Aufgaben im Rahmen der integrierten Versorgung ermittelt. So wird festgestellt, wer im Rahmen des Versorgungsverbunds miteinander kommunizieren soll.

Patterns der Gruppe „Analyse des Kommunikationsbedarfs“:

- Analyse der organisatorischen Verteilung im Verbund (Anhang S. 272)
- Untersuchung betroffener Subsysteme (Anhang S. 595).

Anschließend wird mittels der Untersuchung betroffener Subsysteme ein Katalog der Softwaresysteme, die an den einrichtungsübergreifenden Behandlungspfaden der verteilten Krankenakte beteiligt sein sollen, einschließlich ihrer verfügbaren Schnittstellen und Standards erstellt. Die Untersuchung betroffener Subsysteme setzt neben einer vorherigen Analyse der organisatorischen Verteilung im Verbund auch eine vorher durchgeführte Analyse der betroffenen Behandlungspfade voraus (siehe Abbildung 45).

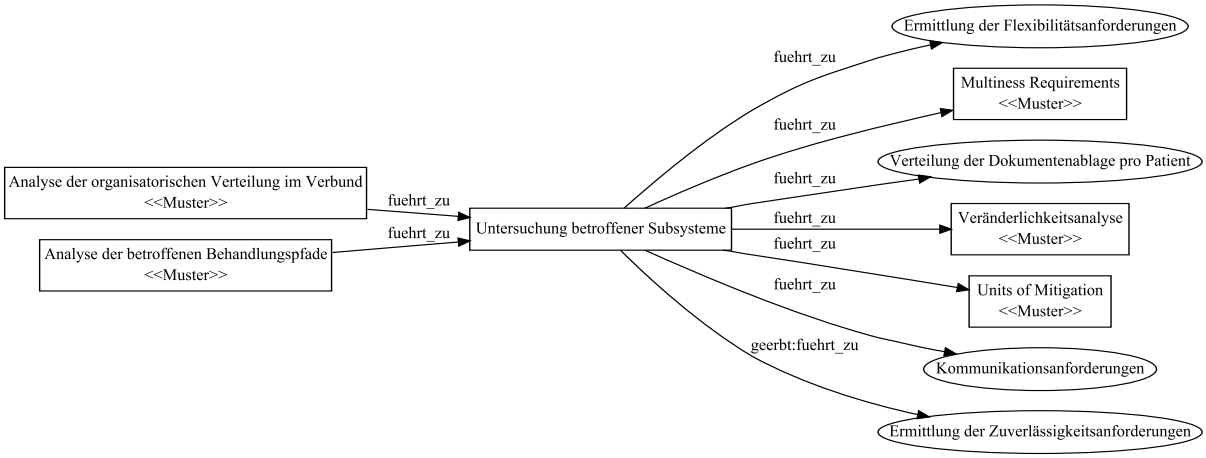



Abbildung 45: „Untersuchung betroffener Subsysteme“ einschließlich direkter Beziehungen

Das Ergebnis der Untersuchung betroffener Subsysteme ist eine Eingangsinformation für verschiedene Patterns der Anforderungsdokumentation und beeinflusst auch direkt die Auswahl von Patterns aus verschiedenen Gruppen der Ebene *Architektur*.

Aufbauend auf den Ergebnissen der Anwendung der Patterns der Gruppe „Analyse des Kommunikationsbedarfs“ können durch die Patterns der Gruppe „Kommunikationsanforderungen“ konkrete Anforderungen für die Gestaltung der Kommunikationsarchitektur der verteilten Krankenakte und die Auswahl konkreter Kommunikationsmechanismen in Form von Patterns der Designebene abgeleitet werden. Dazu enthält die Gruppe die in der folgenden Auflistung dargestellten sieben Patterns.

Patterns der Gruppe „Kommunikationsanforderungen“:

- Availability Requirements (Anhang S. 292)
- Comply-with-Standard Requirements (Anhang S. 323)
- Inter System Interaction Requirements (Anhang S. 410)
- Inter System Interface Requirements (Anhang S. 412)
- Response Time Requirements (Anhang S. 516)
- Scalability Requirements (Anhang S. 541)
- Technology Requirements (Anhang S. 585).

Die Abbildung 46 zeigt in Form eines Graphen die innerhalb der Gruppe vorliegenden Beziehungen. Die Anwendung mancher Patterns setzt voraus, dass auch bestimmte andere Patterns der gleichen Gruppe angewendet werden. So erfordert beispielsweise die Anwendung des Inter-System-Interaction-Requirements-Patterns die vorherige Anwendung des Inter-System-Interface-Requirements-Patterns.

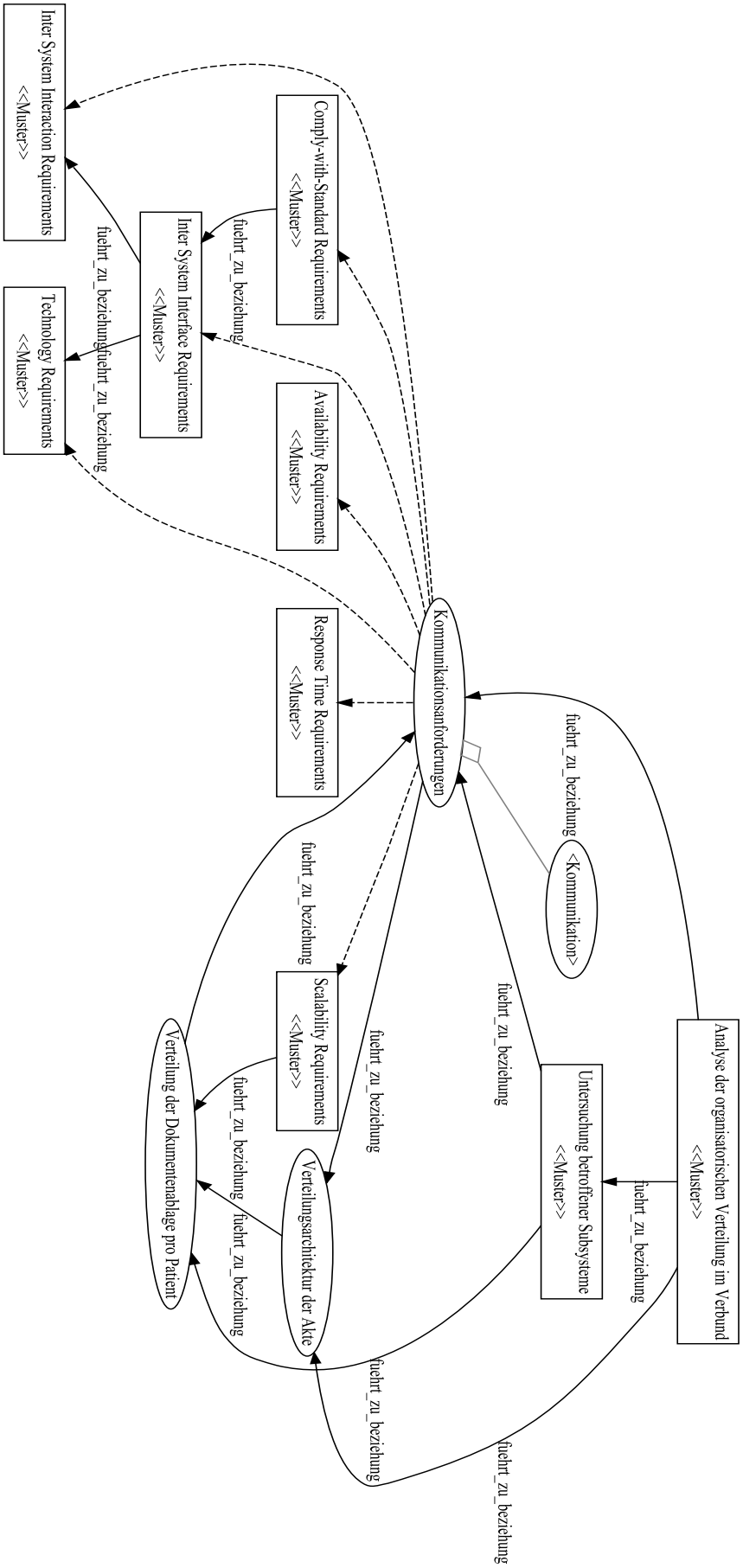



Abbildung 46: Gruppe „Kommunikationsanforderungen“ mit internen Beziehungen

Durch das Inter-System-Interface-Requirements-Pattern werden auf Basis der Ergebnisse der Untersuchung betroffener Subsysteme einzelne Schnittstellen zwischen den beteiligten Subsystemen identifiziert, benannt und detailliert als Anforderung beschrieben. Darauf aufbauend wird durch die Anwendung des Inter-System-Interaction-Requirements-Patterns pro identifiziertem Interface der Zweck und die zur Nutzung des Interfaces notwendige Interaktion beschrieben. Diese Beschreibung ist nur dann möglich, wenn vorher die erforderlichen Schnittstellen identifiziert sind.

Ähnliche Beziehungen existieren auch zwischen anderen Patterns dieser Gruppe und Patterns anderer Gruppen. Beschreibungen zu den in den Graphen abgebildeten Beziehungen befinden sich jeweils im Patternkatalog im Anhang. Dort liegt für jede abgebildete Kante eine textuelle Erläuterung oder ein Verweis auf eine Quelle mit Informationen zu der Beziehung vor.

### 5.2.2. Architektur

Zusätzlich zu den definierten Kommunikationsanforderungen bedarf die Gestaltung der Kommunikationsarchitektur der verteilten Krankenakte einzelner Ergebnisse aus der Anforderungsanalyse der Schwerpunkte *Flexibilität*, *Sicherheit* und *Zuverlässigkeit*. Sie bilden gemeinsam die Basis für die Auswahl geeigneter Patterns aus den Gruppen der Ebene *Architektur* des Schwerpunkts *Kommunikation*.

Gruppen der Ebene *Architektur* im Schwerpunkt *Kommunikation*:

- Verteilungsarchitektur der Akte
- Verteilung der Dokumentenablage pro Patient
- Grundlegende Integrationsformen
- Grundlegende Kommunikationsarten
- Asynchrone Kommunikation
- Synchrone Kommunikation
- Richtung des Datenaustauschs
- Rolle der Kommunikanten.

Die Auswahl einer geeigneten Verteilungsarchitektur ist eine zentrale Architekturentscheidung bei der Entwicklung einer verteilten Krankenakte. Die Verteilungsarchitektur ist sowohl von den ermittelten Rahmenbedingungen aus der organisatorischen Verteilung im Verbund als auch von den relevanten Behandlungspfaden und Subsystemen beeinflusst. Die Patterns zur Gestaltung der Verteilungsarchitektur sind in den Gruppen „Verteilung der Dokumentenablage pro Patient“ und „Verteilungsarchitektur der Akte“ zusammengefasst.

Die Patterns der Gruppe „Verteilungsarchitektur der Akte“ beschreiben drei verschiedene Architekturansätze zur Gestaltung der Ablage von Akteninhalten auf die Knoten der verteilten Krankenakte. Das Spektrum der enthaltenen Patterns reicht von der Akte mit vollständig dezentraler Datenspeicherung, die auch als maximal verteilte Krankenakte bezeichnet werden kann, bis hin zur Akte mit zentraler Datenspeicherung. Die Akte mit zentraler Datenspeicherung erreicht bezüglich des Aspekts der Verteilung die Grenzen der Definition verteilter Krankenakten (siehe Kapitel 2.2), denn es ist möglich mit diesem Pattern auch

Systeme zu gestalten, die nicht der Definition verteilter Krankenakten entsprechen. Eine Zuordnung zu den verteilten Krankenakten ist bei zentraler Datenspeicherung immer dann möglich, wenn mindestens die Erstellung von Dokumenten oder deren Zwischenspeicherung auf dezentralen Systemknoten mit eigenen Softwaresystemen erfolgt oder der zentrale Datenspeicher selbst in Form eines verteilten Systems realisiert ist.

Pattern der Gruppe „Verteilungsarchitektur der Akte“:

- Akte mit regional zentralisierter Datenspeicherung (Anhang S. 257)
- Akte mit vollständig dezentraler Datenspeicherung (Anhang S. 260)
- Akte mit zentraler Datenspeicherung (Anhang S. 264).

Unter einer „Akte mit regional zentralisierter Datenspeicherung“ ist eine Mischform der Patterns „Akte mit vollständig dezentraler Datenspeicherung“ und „Akte mit zentraler Datenspeicherung“ zu verstehen. Während bei der zentralen Datenspeicherung Vorteile durch den reduzierten Aufwand für die Bereitstellung von Schnittstellen und eine erleichterte Administration z. B. durch zentrale Datensicherung und zentrale Konfiguration erreicht werden können, liegen die Vorteile von Akten mit vollständig dezentraler Datenspeicherung in der Beibehaltung existierender Systemlandschaften und der nutzernahen Speicherung der Inhalte mit einer daraus resultierenden verringerten Netzwerklast. Im Pattern-Kandidaten „Akte mit regional zentralisierter Datenspeicherung“ wird versucht, die Vorteile beider Ansätze durch den gebietsweisen Aufbau zentraler Speicher zu kombinieren, die als Verbund einer dadurch deutlich geringeren Zahl dezentraler Speicher agieren.

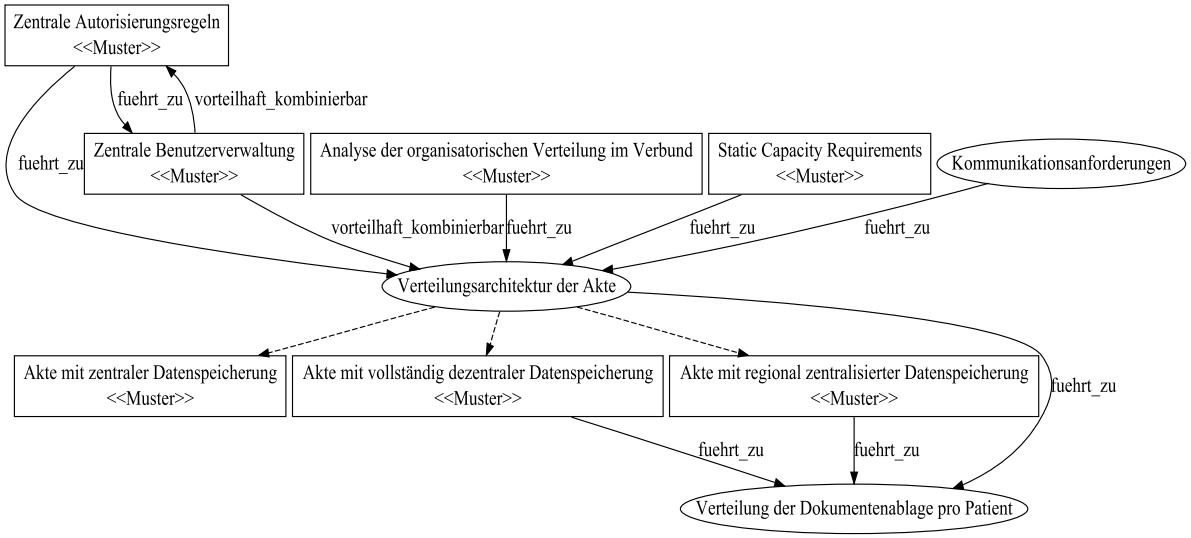



Abbildung 47: Ausgewählte Beziehungen der Gruppe „Verteilungsarchitektur der Akte“

Die Abbildung 47 zeigt den Einfluss verschiedener Patterns auf die Auswahl einer geeigneten Verteilungsarchitektur. Dabei zeigt sich, dass die Auswahl der Verteilungsarchitektur neben den bekannten, bereits erwähnten Faktoren *organisatorische Verteilung* und *Kommunikationsanforderungen* auch von den Static Capacity Requirements (Anhang S. 581), also den Anforderungen an die Speicherkapazität des Gesamtsystems, und von Architekturentscheidungen des Schwerpunkts *Sicherheit* abhängig ist.

Bergmann (Bergmann 2006:S. 53–56) beschreibt mit seiner Sammlung von Entwurfsmustern aus Sicht des Integrationsmodells einen Ansatz, der der Gruppe „Verteilungsarchitektur der Akte“ ähnlich ist. Die enthaltenen Muster vermischen allerdings die Aspekte der Verteilung der Datenspeicherung und der grundlegenden Integrationsformen. Mit der ebenfalls enthaltenen Betrachtung nach dem Zeitpunkt der Bereitstellung der Daten wird außerdem der Aspekt der Richtung des Datenaustauschs in die gleiche Gruppe von vier Patterns integriert. Alle vier von Bergmann beschriebenen Patterns lassen sich durch gezielte Kombination der Patterns aus den genannten Gruppen bilden. Das lässt sich am Beispiel des Musters „Integration zum Zeitpunkt der Bereitstellung“ (Bergmann 2006:S. 53 Abbildung 10) erläutern. Es beschreibt eine spezielle Form der Datenbereitstellung mittels des Push-Prinzips, bei der entweder in eine Akte mit zentraler oder regional zentralisierter Datenspeicherung (als Senke) Daten bereitgestellt werden. Das Muster kann, je nach konkreter Umsetzung auch der grundlegenden Integrationsform *Shared Database* entsprechen.

Außerdem gibt es auch zwischen der von Bergmann (Bergmann 2006:S. 47–48) dargestellten Verteilung und Redundanz nach Beß und den hier vorgestellten Konzepten inhaltliche Überlappungen. Die dort beschriebene Klassifikation von Verteilung und Redundanz zerlegt das Feld der elektronischen Gesundheitsakten (EGA) aber nach anderen Kriterien, als das bei der Gruppe „Verteilungsarchitektur“ der Fall ist. So wird z. B. zusätzlich eine Gliederung nach stationären und mobilen EGA eingeführt. Hierbei handelt es sich um ein Kriterium, das für die Patterns der Gruppe „Verteilungsarchitektur“ aus technischen Gründen nicht relevant ist.

Neben den internen Beziehungen der Gruppe „Verteilungsarchitektur der Akte“ zeigt die Abbildung 47 auch die direkten Beziehungen der Gruppe „Verteilungsarchitektur der Akte“ und ihrer enthaltenen Patterns zur Gruppe „Verteilung der Dokumentenablage pro Patient“. Diese Gruppe umfasst zwei Patterns für die Gestaltung des Ablageprinzips von Dokumenten bezogen auf je einen einzelnen Patienten.

Patterns der Gruppe „Verteilung der Dokumentenablage pro Patient“:

- Fully Distributed Patient Documents (Anhang S. 394)
- Single Storage-Point per Patient (Anhang S. 568).

Die Patterns der Gruppe „Verteilung der Dokumentenablage pro Patient“ sind mit den Patterns der Gruppe „Verteilungsarchitektur der Akte“ nahezu frei kombinierbar. Lediglich eine Kombination des Patterns „Fully Distributed Patient Documents“ mit dem Pattern „Akte mit zentraler Datenspeicherung“ ist nicht möglich, da bei einer zentralen Speicherung aller Daten keine Verteilung der Dokumente eines einzelnen Patienten auf verschiedene Knoten möglich ist.

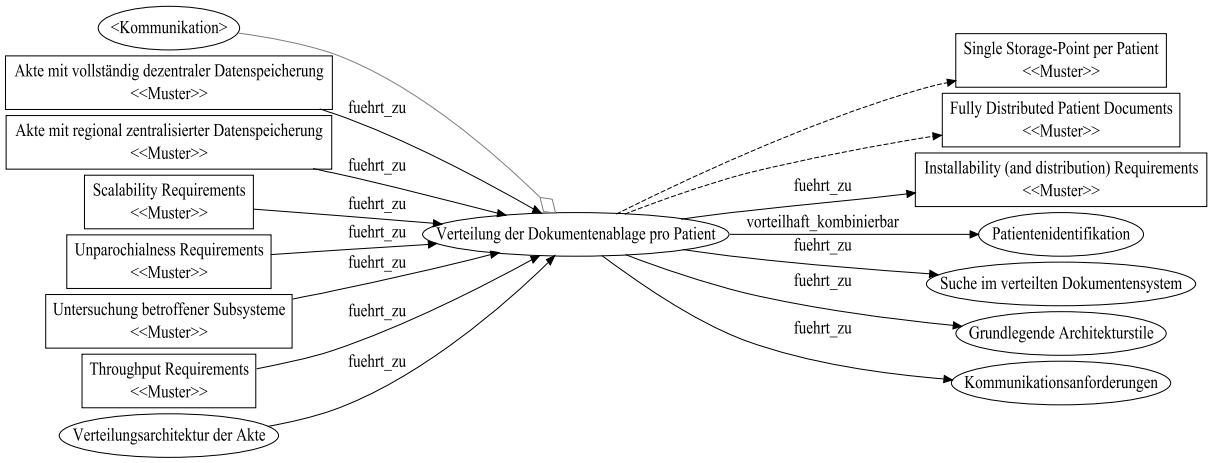



Abbildung 48: Direkte Beziehungen der Gruppe „Verteilung der Dokumentenablage pro Patient“

Die Auswahl der Verteilung der Dokumentenablage pro Patient wird, wie in Abbildung 48 dargestellt, von einer Vielzahl anderer Patterns beeinflusst. Neben der Gruppe „Verteilungsarchitektur“ und ihren Patterns handelt es sich um verschiedene Patterns der Anforderungsanalyse. Die Dokumentenablage pro Patient beeinflusst die Skalierbarkeit und den möglichen Durchsatz des Systems aber auch dessen Anpassbarkeit an verschiedene Umgebungssituationen. Aus diesem Grund sind die entsprechenden Anforderungen auch bei der Auswahl des Patterns bezüglich der Dokumentenablage pro Patient relevant.

In ausgehender Richtung beeinflusst das ausgewählte Pattern der Verteilung der Dokumentenablage pro Patient auch die Auswahl einer Menge anderer Patterns aus den Gruppen „Patientenidentifikation“, „Suche im verteilten Dokumentensystem“ und „Grundlegende Architekturstile“. Zusätzlich zu den bisher erwähnten Einflüssen resultieren aus der Auswahl auch weiter Anforderungen, die auf den eigenen Schwerpunkt und fremde Schwerpunkte wirken.

Neben der Auswahl einer geeigneten, von domänenspezifischen Fragestellungen geprägten Verteilungsarchitektur für die verteilte Krankenakte gilt es, auch bei den grundlegenden Integrationsformen ein oder mehrere geeignete Patterns auszuwählen. Die grundlegenden Integrationsformen sind nicht spezifisch für verteilte Krankenakten, sondern für beliebige Softwaresysteme verwendbar. Sie beschreiben erprobte Ansätze mit denen mehrere, eigentlich eigenständige Softwaresysteme zur Zusammenarbeit befähigt werden können.

Patterns der Gruppe „Grundlegende Integrationsformen“:

- Data Replication (Anhang S. 335)
- Distributed Business Process (Anhang S. 346)
- File Transfer (Anhang S. 383)
- Information Portal (Anhang S. 405)
- Service-oriented Architecture (Anhang S. 555)
- Shared Business Function (Anhang S. 559).

Die Auswahl eines geeigneten Patterns der Gruppe „Grundlegender Integrationsformen“ wird durch die Ergebnisse der Patterns „Multiness Requirements“, „Throughput Requirements“,

„Inter System Interaction Requirements“ und „Comply-with-Standard Requirements“ beeinflusst (siehe Abbildung 49). Detaillierte Erläuterungen zu den Einflüssen dieser Patterns auf die Auswahl können in dem im Anhang enthaltenen Patternkatalog den jeweiligen Beschreibungen der Patterns entnommen werden.

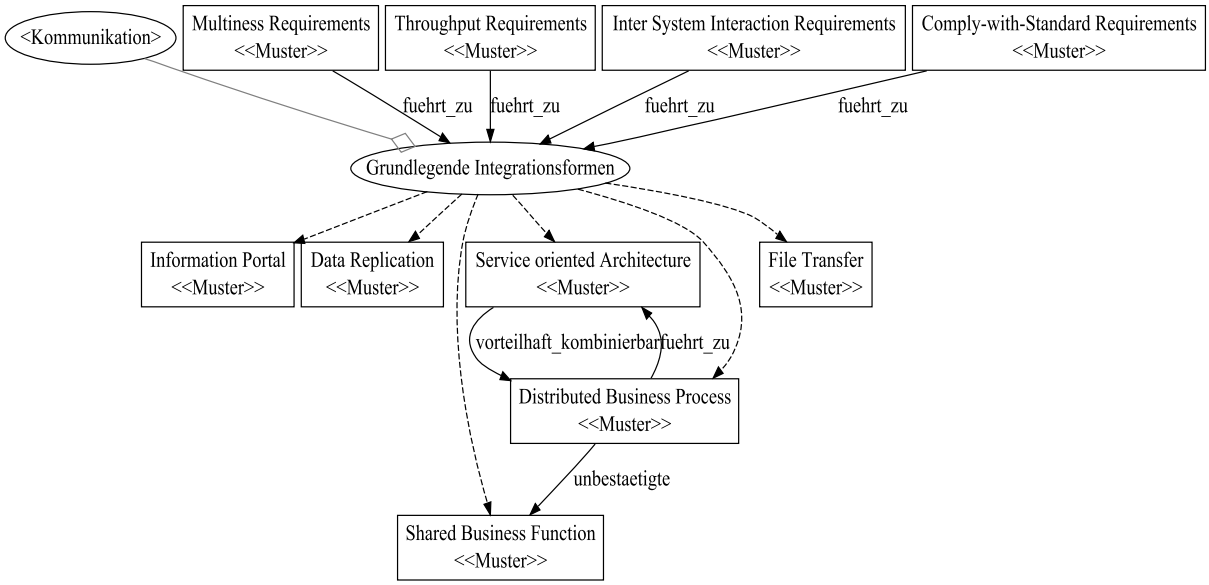



Abbildung 49: Gruppe „Grundlegende Integrationsformen“ mit internen Abhängigkeiten

Die Patterns zu den grundlegenden Integrationsformen umfassen verschiedene Muster, die Integration mit unterschiedlicher Intensität und unterschiedlichen technologischen Ansätzen ermöglichen. Die Integrationsform „Information Portal“ beschreibt eine Form der Integration, die ausschließlich auf Ebene der Benutzeroberfläche (UI) stattfindet. Das setzt integrierbare UIs oder Shared Business Functions für die Erstellung integrierbarer UIs voraus. Die Data Replication stellt einen Ansatz dar, der eine Integration auf Ebene der Datenspeicherung, also z. B. der Datenbank ermöglicht. Die notwendigen Schnittstellen werden auf der Ebene der Datenspeicherung geschaffen. Änderungen an gespeicherten Daten einer Applikation werden in den Datenspeicher von anderen Applikationen repliziert. Der Integrationsansatz „Shared Business Function“ ermöglicht wie auch der Ansatz „Service-oriented Architecture“ eine Integration auf der Ebene der Anwendungslogik. Dabei werden remotefähige Funktionen oder Methoden, die Aufgaben der Geschäftslogik abbilden, von verschiedenen Applikationen gemeinsam genutzt. Solche remotefähigen Funktionen oder Methoden sind hilfreich und teils sogar notwendig, um mit dem Distributed-Business-Process-Pattern eine Integration auf der Ebene von Prozessbestandteilen verschiedener Applikationen zu einem kohärenten Prozess zu schaffen.

Die verschiedenen grundlegenden Integrationsformen basieren auf unterschiedlichen grundlegenden Kommunikationsarten oder verwenden diese. Die Kommunikationsarten sind im vorliegenden Patternkatalog in die Gruppen „Richtung des Datenaustauschs“ und „Rolle der Kommunikanten“ gegliedert.

Inhalt der Gruppe „Grundlegende Kommunikationsarten“:

- Gruppe: Richtung des Datenaustauschs

- Gruppe: Rolle der Kommunikanten.

Die Gruppe „Richtung des Datenaustauschs“ enthält zwei in Patternform beschriebene Grundprinzipien des Datenaustauschs. Sie erläutern die beiden Formen des Datenaustauschs aus der Sicht des Akteurs bzw. des Prozesses, der den Datenaustausch initiiert.

Patterns der Gruppe „Richtung des Datenaustauschs“:

- Pull-Prinzip (Anhang S. 492)
- Push-Prinzip (Anhang S. 494)

Das Pull-Prinzip beschreibt das aktive Abholen von Daten bei einer Senke durch den konsumierenden Prozess oder Akteur. Das bedeutet, dass das Subjekt, das Daten benötigt, um diese zu verwenden, diese Daten aktiv abholt. Das Push-Prinzip beschreibt hingegen die aktive Übermittlung von Daten durch den Prozess oder Akteur, der die Daten erzeugt hat oder aus anderen Gründen bereits besitzt, an den konsumierenden Prozess oder Akteur.

Details zu den Prinzipien *Push* und *Pull* sind in den entsprechenden Pattern-Beschreibungen im Patternkatalog sowie in Kapitel 2.1.8 auf S. 26 erläutert.

Die zweite Untergruppe innerhalb der Gruppe „Grundlegende Kommunikationsarten“ ist die Gruppe „Rolle der Kommunikanten“. Die beiden enthaltenen Patterns beschreiben zwei grundlegende Prinzipien zur Verteilung von Aufgaben in verteilten Systemen.

Patterns der Gruppe „Rolle der Kommunikanten“:

- Client-Server (Anhang S. 318)
- Peer to Peer (Anhang S. 469).

Während bei Client-Server-Systemen eine klare Rollenverteilung zwischen Dienstkonsumenten und Dienstanbietern vorliegt, treten bei Peer-to-Peer-Systemen alle Knoten gleichberechtigt, sowohl in der Rolle des Dienstanbieters als auch des Dienstkonsumenten auf. Außerdem erfolgt die Kommunikation in Peer-to-Peer-Systemen direkt zwischen den Peers (Knoten) und nicht über Kommunikations-vermittelnde Server.

Neben den bisher genannten Ansätzen kann die Form der Kommunikation auch durch die Aspekte *Direktheit* und *Zeit* kategorisiert werden. Die Frage, ob eine Nachricht direkt einem Empfänger zugestellt und ob ggf. sogar direkt auf eine Antwortnachricht gewartet wird, unterteilt die Kommunikation in die Formen *synchrone* und *asynchrone Kommunikation*. Ausführliche Erläuterungen zu synchroner und asynchroner Kommunikation befinden sich im Kapitel 2.1.8 in der einführenden Erläuterung zur Kommunikation in verteilten Systemen.

Patterns der Gruppe „Asynchrone Kommunikation“:

- Messaging (Anhang S. 441)
- Point to Point Messaging (Anhang S. 473)

- Publish-Subscribe Messaging (Anhang S. 489).

Das Pattern „Messaging“ beschreibt allgemein den Aufbau von Softwaresystemen, die durch asynchron übermittelte Nachrichten (vgl. Hohpe & Woolf 2010:S. 53) lose aneinander gekoppelt und so zu einem gemeinsamen System integriert sind. Es wird, wie in Abbildung 50 dargestellt, durch die Patterns „Publish-Subscribe Messaging“ und „Point to Point Messaging“ spezialisiert.

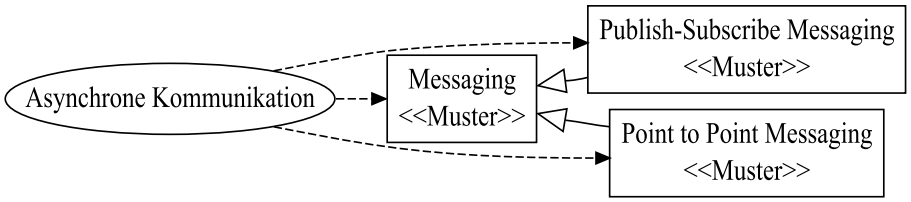



Abbildung 50: Vererbungsbeziehung in der Gruppe „Asynchrone Kommunikation“

Das Pattern „Point to Point Messaging“, das auch unter dem Namen „Point-to-Point Channel“ (Hohpe & Woolf 2010:S. 103) bekannt ist, beschreibt ein Vorgehen, bei dem eine Nachricht von einem Sender zu einem Empfänger unter Zuhilfenahme einer Nachrichten-Warteschlange (Queue) asynchron übermittelt wird. Das Pattern „Publish-Subscribe Messaging“ erläutert eine Form der nachrichtenorientierten Kommunikation, bei der eine Menge registrierter Empfänger Nachrichten aus einem Verteilungspunkt oder zu einem Thema erhält. Es ist auch unter dem Namen „Publish-Subscribe Channel“ (Hohpe & Woolf 2010:S. 106) bekannt.

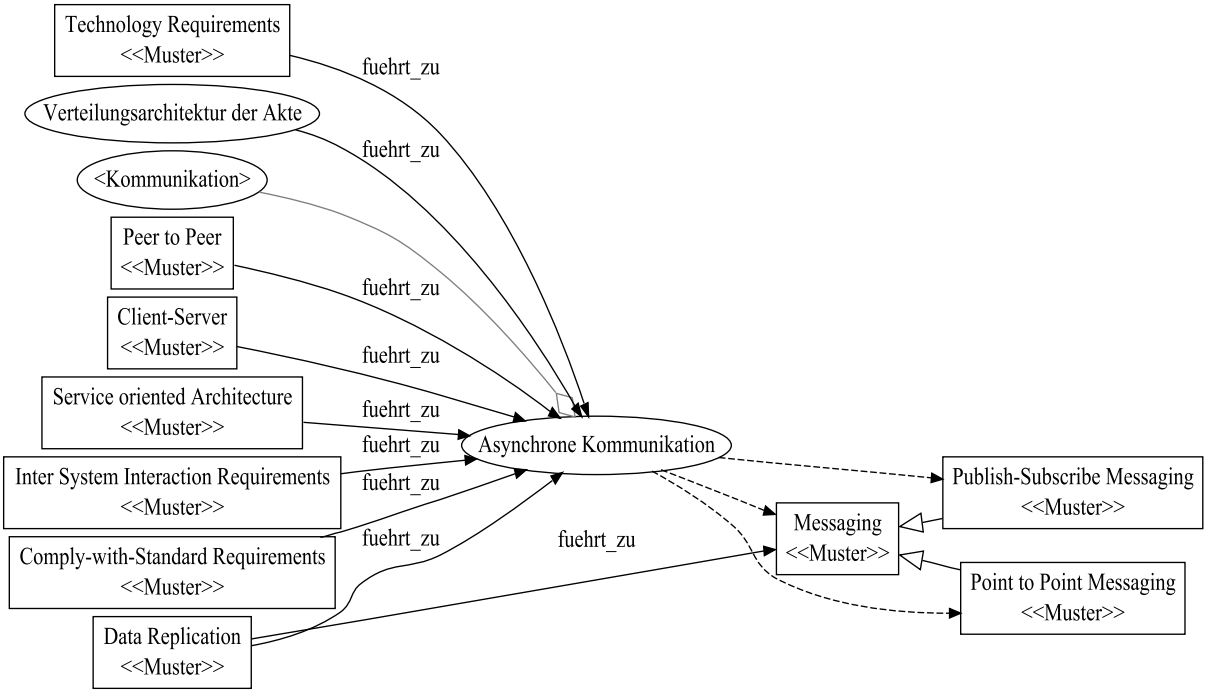



Abbildung 51: Gruppe „Asynchrone Kommunikation“ mit direkten Beziehungen

Die beiden Patterns „Publish-Subscribe Messaging“ und „Point to Point Messaging“ zeigen grundlegende Architekturstile für nachrichtenorientierte Kommunikation. Quellen wie das Buch „Enterprise Integration Patterns“ (Hohpe & Woolf 2010) von Hohpe und Woolf enthalten eine Menge spezieller Design-Patterns für die konkrete Umsetzung und

Unterstützung solcher Architekturen. Diese Patterns sind aufgrund ihres besonders spezifischen Gegenstandsbereichs in der vorliegenden Version nicht in den Kernbestandteil der Pattern-Sprache für verteilte Krankenakten aufgenommen. Sie können im Zuge der iterativen Weiterentwicklung zu einer vollständigen Pattern-Sprache den Quellen entnommen und der Pattern-Sprache erweiternd hinzugefügt werden.

Die Abbildung 51 zeigt, dass die Entscheidung über die Verwendung asynchroner Kommunikation, aber auch die Auswahl eines geeigneten asynchronen Kommunikationsmechanismus durch eine Vielzahl von Faktoren beeinflusst wird. So beeinflussen z. B. Patterns der Anforderungsanalyse wie das Technology-Requirements-Pattern, das Inter-System-Interaction-Requirements-Pattern und das Comply-with-Standard-Requirements-Pattern die Auswahl asynchroner Mechanismen ebenso wie eine bereits gewählte Verteilungsarchitektur der Akte.

Das logische Gegenteil zu asynchroner Kommunikation ist die synchrone Kommunikation. Auch ihre Eigenschaften werden in Kapitel 2.1.8 „Einführung in die Kommunikation in verteilten Systemen“ (S. 24) erläutert. Ein wesentlicher Teil der entfernten Kommunikation zwischen Softwaresystemen läuft synchron ab. Synchrone und asynchrone Kommunikationsmechanismen schließen sich innerhalb einer Systemarchitektur nicht gegenseitig aus. Problemadäquat verwendet, können sie einander sinnvoll ergänzen.

Patterns der Gruppe „Synchrone Kommunikation“:

- Remote Procedure Invocation (Anhang S. 509)
- Shared Database (Anhang S. 561).

Die Gruppe „Synchrone Kommunikation“ enthält in ihrem dargestellten Zustand die zwei aufgelisteten Patterns und ist als solche noch unvollständig. Das bedeutet, dass die Gruppe möglicherweise durch Hinzufügen weiterer Patterns verbessert werden kann. Während das Pattern „Remote Procedure Invocation“ ein klassischer Vertreter synchroner Kommunikation ist, ist die Zuordnung des Integrationspatterns „Shared Database“ von der Betrachtung der zugrunde liegenden Kommunikation abhängig. Die Kommunikation zwischen den verschiedenen Systemen und der gemeinsam genutzten Datenbank erfolgt immer synchron. Aber die Kommunikation zwischen den beteiligten Systemen, die über die gemeinsame Datenbank erfolgt, kann auch asynchron abgewickelt werden. Das ist z. B. dann möglich, wenn eine Tabelle in der gemeinsam genutzten Datenbank zur Ablage einer Nachrichten-Warteschlange verwendet wird. Beim klassischen Ansatz der Shared Database wird allerdings, wie von Fowler (Hohpe & Woolf 2010:S. 47) beschrieben, auch diese Kommunikation durch direktes Ändern der vom Kommunikationspartner verwendeten Daten, mittels eines synchronen Aufrufs an der Datenbank, abgewickelt. Aus diesem Grund ist das Pattern „Shared Database“ der Gruppe „Synchrone Kommunikation“ zugeordnet.

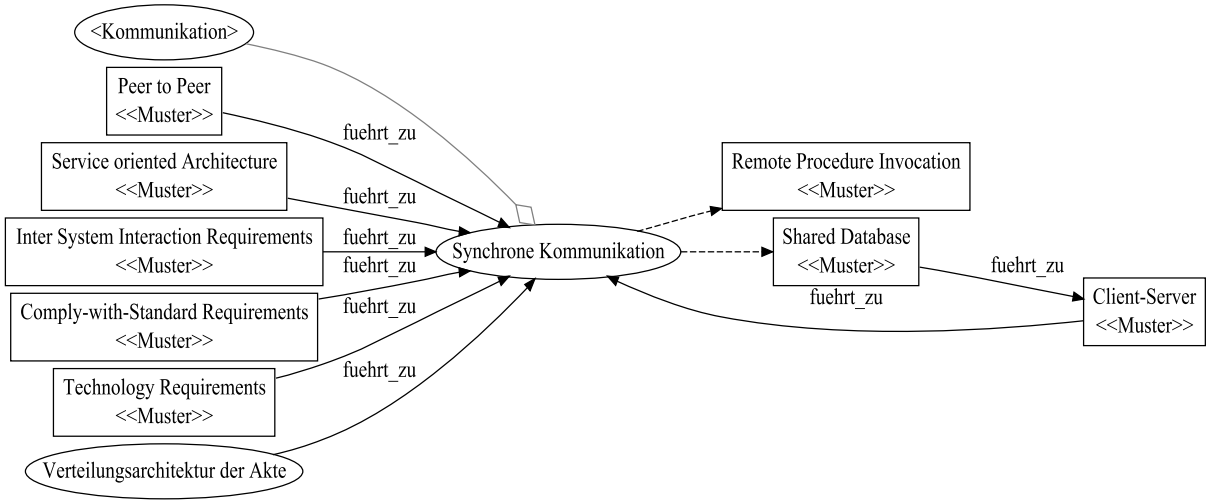



Abbildung 52: Die Gruppe „Synchrone Kommunikation“ mit internen Beziehungen

Die Abbildung 52 zeigt die Gruppe „Synchrone Kommunikation“ mit ihren Patterns und ihren direkten Beziehungen. Dabei zeigt die Abbildung, dass sowohl die Ergebnisse der Anwendung von verschiedenen Patterns der Anforderungsanalyse als auch die Auswahl bestimmter Architekturpatterns des Schwerpunkts *Kommunikation* zur Verwendung synchroner Kommunikationsmechanismen führen. Das Integrationspattern „Shared Database“ führt aufgrund seiner eindeutigen Zuordnung der Datenbank zur Rolle Server zu einer zwangsläufigen Adaption des Client-Server-Patterns, das gegebenenfalls auch auf einer mehrschichtig verteilten Architektur aufgebaut werden kann.

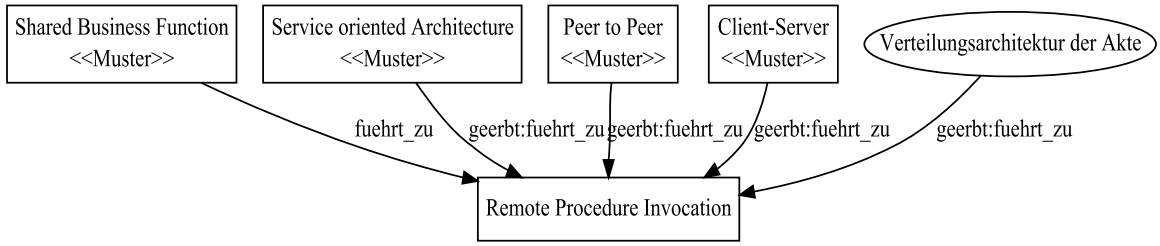



Abbildung 53: Eine Auswahl eingehender Beziehungen des Patterns „Remote Procedure Invocation“

Wie bereits erläutert, kann das Remote-Procedure-Invocation-Pattern als typischer Vertreter der Gruppe „Synchrone Kommunikation“ betrachtet werden. Es wird zur Umsetzung von verschiedenen Integrationsformen verwendet. So ist es beim Einsatz der Integrationsformen „Shared Business Function“ oder „Service-oriented Architecture“ notwendig, große Teile der zugrunde liegenden Kommunikation mittels des Remote-Procedure-Invocation-Patterns zu konzipieren. Außerdem zeigt die Abbildung 53 auch, dass bei der Umsetzung von Software in Form von Peer-to-Peer- oder Client-Server-Architekturen (Gruppe: Rolle der Kommunikanten) synchrone Kommunikationsmechanismen, z. B. in Form des Patterns „Remote Procedure Invocation“, verwendet werden können oder sogar müssen.

Zur Umsetzung des Patterns „Remote Procedure Invocation“ selbst wird, wie in Abbildung 54 dargestellt, primär auf vier grundlegende Design-Patterns zurückgegriffen.

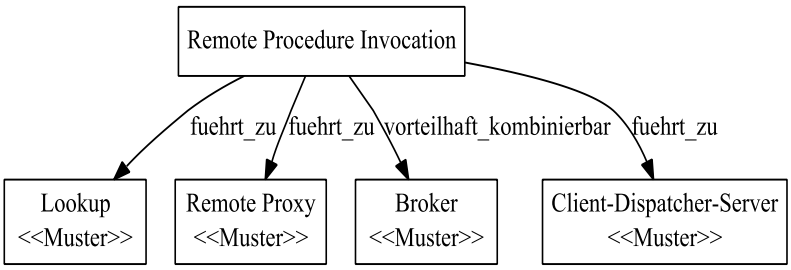



Abbildung 54: Übergang vom Architekturpattern „Remote Procedure Invocation“ zu den Designpatterns

Diese vier Patterns werden im nächsten Kapitel erläutert und bilden die Basis für die transparente Realisierung der Kommunikation in verteilten Systemen. Dabei finden sie sowohl bei der Realisierung synchroner als auch asynchroner Kommunikationsmechanismen Verwendung. Ein Schwerpunkt der Anwendung ist jedoch die Umsetzung von synchronen Kommunikationsmechanismen wie der Remote Procedure Invocation.

### 5.2.3. Design

Auf Ebene des Softwaredesigns kann die Kommunikation durch eine Vielzahl von Patterns gestaltet werden, die die Flexibilität, Sicherheit und Zuverlässigkeit der kommunizierenden Anwendungen und der Kommunikation selbst beeinflussen. Der größte Teil dieser Patterns ist in den Gruppen der Ebene *Design* innerhalb der Schwerpunkte *Flexibilität*, *Sicherheit* und *Zuverlässigkeit* zu finden. Ein spezieller Aspekt, der zwar auch die Flexibilität des die Kommunikation betreffenden Quellcodes verbessert, der aber primär der Erreichung der Kommunikationsfähigkeit in den Softwarekomponenten dient, ist das Verbergen der Kommunikation vor dem Anwendungsentwickler. Aus diesem Grund enthält die Ebene *Design* des Schwerpunkts *Kommunikation* im Kernbestandteil der Pattern-Sprache ausschließlich die Gruppe verbergen der Kommunikation. Die Ebene kann allerdings, wie alle Schwerpunkte und Ebenen, zur Vervollständigung der Pattern-Sprache um weitere Gruppen erweitert werden.

Patterns der Gruppe „Verbergen der Kommunikation“:

- Broker (Anhang S. 300)
- Client-Dispatcher-Server (Anhang S. 317)
- Lookup (Anhang S. 429)
- Remote-Proxy (Anhang S. 512).

Die vier Patterns der Gruppe „Verbergen der Kommunikation“ dienen der Kapselung und Vereinfachung der Kommunikation zwischen verteilten Softwarekomponenten. Das Broker-Pattern – in der Literatur (Buschmann u. a. 2007a:S. 237; Buschmann u. a. 1996:S. 99) oft als Architekturpattern kategorisiert – wird hier der Ebene *Design* zugeordnet, da es nach der Analyse der verschiedenen Beziehung eine gemeinsame Gruppe mit den grundsätzlich (Gamma u. a. 2010:S. 207 ff.; Buschmann u. a. 1996:S. 263 und 323) als Design-Patterns identifizierten Patterns „Remote-Proxy“, „Lookup“ und „Client-Dispatcher-Server“ bildet. Die client- und serverseitige Broker-Implementierung kapseln die Aspekte der Kommunikation. Sie erfüllen die Aufgaben des Serialisierens und Deserialisierens sowie der Übermittlung von Nachrichten an den jeweils geeigneten Kommunikationspartner. Kombiniert mit dem

Remote-Proxy-Pattern (siehe Buschmann u. a. 2007a:S. 237, Abbildung unten) kann dem Entwickler der Client-Software ein lokaler, der entfernten Komponente schnittstellengleicher Stellvertreter zur Verfügung gestellt werden. Im Umkehrschluss kann, wie in Abbildung 55 dargestellt, der für den Remote-Proxy erforderliche Code durch die gleichzeitige Anwendung des Broker-Patterns vereinfacht und verkürzt werden.

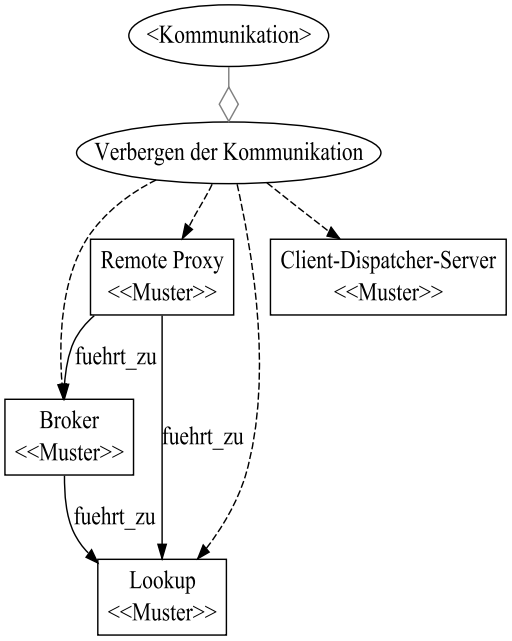



Abbildung 55: Patterns der Gruppe „Verbergen der Kommunikation“ mit internen Beziehungen

Die Verwendung des Lookup-Patterns ermöglicht die ortstransparente Nutzung entfernter Komponenten. Das Pattern beschreibt die Adressierung von Komponenten im verteilten System unter Nutzung symbolischer Namen. Der symbolische Name der Komponente wird z. B. zum Zeitpunkt der Initialisierung eines Proxy-Objekts aufgelöst, sodass für den Remote-Proxy die Adresse des realen Ablageorts der entfernten Komponente bekannt wird. Für die Programmierung und die Initialisierung des Proxy im Code der konsumierenden Komponente muss die reale Adresse der entfernten Komponente zum Zeitpunkt der Programmierung nicht bekannt sein.

Das Pattern „Client-Dispatcher-Server“ (siehe Buschmann u. a. 1996:S. 323) zeigt einen anderen Weg zur Erreichung von Ortstransparenz bei der Entwicklung von Systemen mit verteilt vorgehaltenen Diensten. Die Dispatcher-Komponente vermittelt die Anfragen des Clients an den oder die jeweils passenden Server.

## *5.3. Sicherheit*

Die Fähigkeit zur Kommunikation ist die zentrale Voraussetzung für das Funktionieren eines verteilten Systems. Für den praktischen Einsatz eines verteilten Systems ist allerdings neben der Fähigkeit zur Kommunikation auch deren Vertraulichkeit und Sicherheit notwendig. Bei einer verteilten Krankenakte, einem medizinischen Informationssystem, das als verteiltes System konzipiert ist, resultieren aus der damit verbundenen Verarbeitung medizinischer Daten besondere Ansprüche an die Vertraulichkeit und Zuverlässigkeit der Daten (vgl.

Kapitel 2.4.2 u. Haas 2009:S. 477). Deshalb spielt der Aspekt der Sicherheit bei der Konzeption und Umsetzung des zugehörigen Softwaresystems eine wichtige Rolle.

Wie die beiden bereits dargestellten Schwerpunkte *Flexibilität* und *Kommunikation* gliedert sich auch der Schwerpunkt *Sicherheit* in die drei Ebenen *Anforderungsanalyse*, *Architektur* und *Design*. Diese Darstellungsform und die Beziehungen der enthaltenen Patterns machen die strukturierte Sammlung von Patterns des Schwerpunkts *Sicherheit* zu einem, den Planungs- und Umsetzungsprozess sicherheitsrelevanter Bestandteile des Softwaresystems konkreter verteilter Krankenakten leitenden Hilfsmittel.

### 5.3.1. Anforderungsanalyse

Im Rahmen der Anforderungsanalyse wird die Sicherheitssituation für das Gesamtsystem analysiert. Hierbei sind sowohl die fachlich bedingten Sicherheitsbedarfe, als auch die existierenden Risiken zu berücksichtigen. Aufbauend auf der umfassenden Analyse aller die Sicherheit des Systems beeinflussenden Kriterien, werden als Ergebnis der Phase *Anforderungsanalyse* konkrete Sicherheitsanforderungen definiert. Die konkreten Sicherheitsanforderungen stellen aufgrund ihrer Abhängigkeiten die Verbindung zu konkreten Mustern oder Gruppen von Patterns in der Ebene *Architektur* her.

Der Schwerpunkt *Sicherheit* enthält in der Ebene *Anforderungsanalyse* neben dem keiner Gruppe explizit zugeordneten Pattern „Security needs Identification for Enterprise Assets“ (Anhang S. 551) vier Pattern-Gruppen.

Einstiegspattern in die Anforderungsanalyse des Schwerpunkts *Sicherheit*:

- Security needs Identification for Enterprise Assets (Anhang S. 551).

Gruppen des Schwerpunkts *Sicherheit* in der Ebene *Anforderungsanalyse*:

- Analyse der Sicherheitssituation
- Bewertung der Risikosituation
- Analyse von Sicherheitsproblemen und Lösungsansätzen
- Ermittlung der Sicherheitsanforderungen.

Das Pattern „Security needs Identification for Enterprise Assets“ dient als Einstiegspunkt in die Patternauswahl für die sicherheitstechnische Gestaltung verteilter Krankenakten. Seine Autoren beschreiben es als das Basispattern für alle zu erwartenden Sicherheitsfragen eines Unternehmens. Es hilft die Bereiche zu identifizieren, in denen Sicherheit tatsächlich benötigt wird (Schumacher u. a. 2005:S. 89). Dadurch grenzt es die Menge der sicherheitsrelevanten Systembestandteile ein und ermöglicht so eine detaillierte Analyse der relevanten Bestandteile.

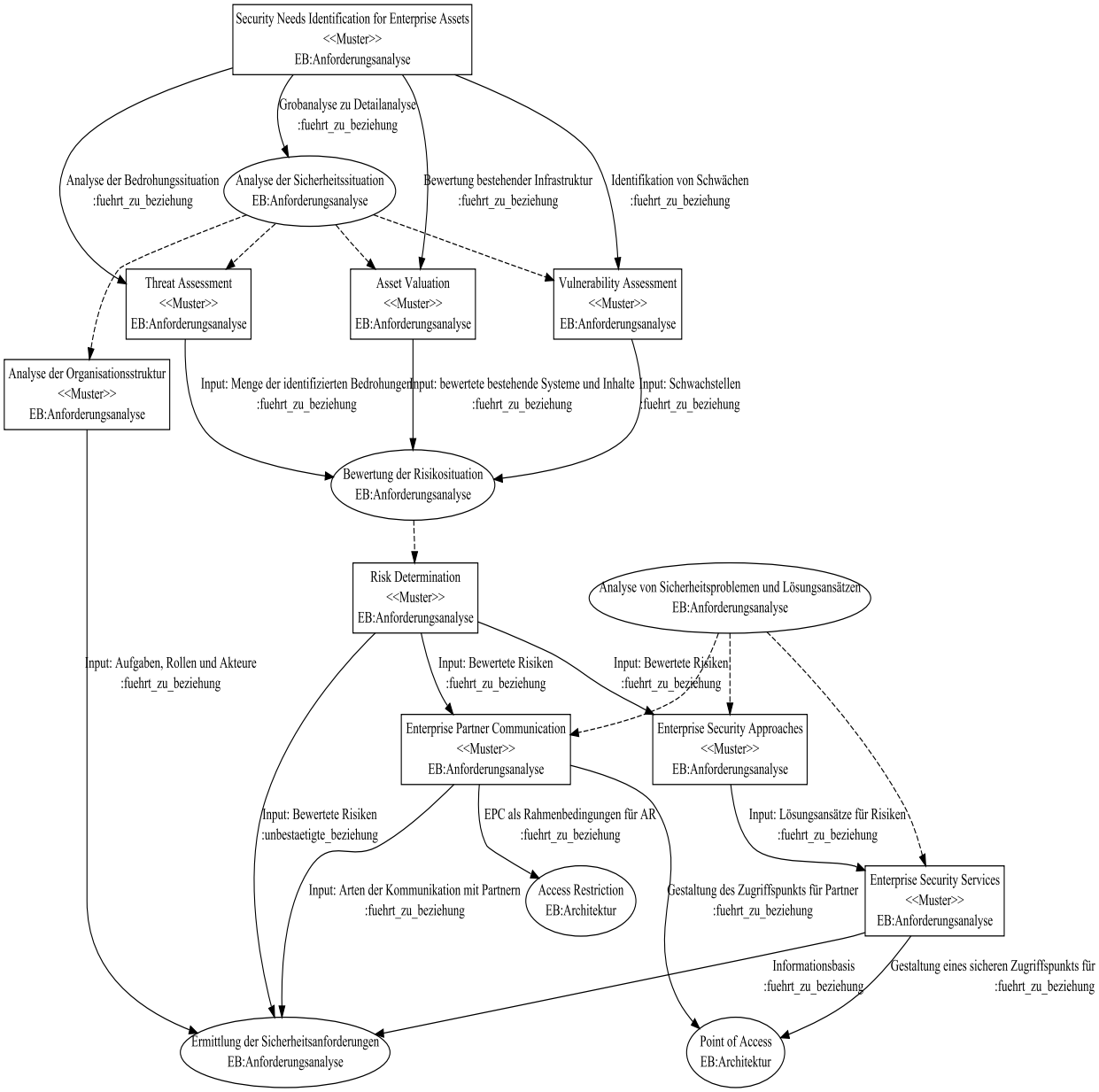



Abbildung 56: Ausschnitt aus der Gesamtsicht der Ebene Anforderungsanalyse im Schwerpunkt Sicherheit

Zur detaillierten Analyse sicherheitsrelevanter Aspekte des Gesamtsystems dienen, wie in Abbildung 56 dargestellt, die vier Patterns der Gruppe „Analyse der Sicherheitssituation“. Sie bauen auf den Ergebnissen der Anwendung des Patterns „Security needs Identification for Enterprise Assets“ auf und liefern jeweils zu einer definierten Menge von Aspekten detaillierte Aussagen zur Sicherheitssituation im fachlichen und organisatorischen Anwendungsbereich des zu entwickelnden Softwaresystems.

Patterns der Gruppe „Analyse der Sicherheitssituation“:

- Analyse der Organisationsstruktur (Anhang S. 270)
- Asset Valuation (Anhang S. 277)
- Threat Assessment (Anhang S. 588)
- Vulnerability Assessment (Anhang S. 604).

Das Pattern „Analyse der Organisationsstruktur“ baut auf den Informationen aus der Anwendung der Patterns „Analyse der betroffenen Behandlungspfade“ (Anhang S. 267) und „Analyse der organisatorischen Verteilung im Verbund“ (Anhang S. 272) auf. Es stellt dadurch eine Verbindung zu den Ergebnissen der Anforderungsanalyse aus den Schwerpunkten *Flexibilität*, *Kommunikation* und *Zuverlässigkeit* her. Eine verteilte Krankenakte ist ein verteiltes System, bei dem die Verteilung der Knoten durch fachliche Einflüsse auch von der organisatorischen Verteilung und den zu erfüllenden Behandlungspfaden abhängig ist. Aus der Analyse der Faktoren *organisatorische Verteilung* und *Behandlungspfade* ergeben sich die Systembeteiligten (Akteure) und deren konkrete Aufgaben im Rahmen der Systemnutzung sowie die organisatorische Zuordnung der Akteure und Aufgaben. Diese Informationen dienen als Basisinformation für die Ermittlung der Sicherheitsanforderungen und für die Definition von Rollen im Sicherheitskonzept. Zudem können die ermittelten Akteure, Aufgaben und Organisationseinheiten für die Bewertung der Gefährdung des Systems durch die Systembeteiligten verwendet werden. Beispielhaft ist das im Rahmen einer speziell angewandten Form des Patterns „Thread Assessment“ möglich.

Das ebenfalls in der Gruppe „Analyse der Sicherheitssituation“ befindliche Pattern „Asset Valuation“, wörtlich übersetzt „Vermögensbewertung“ oder „Anlagenbewertung“, beschreibt ein Verfahren zur Bewertung der Systembestandteile und Systeminhalte bezüglich ihres Sicherungsbedarfs. Wertvollere Inhalte bedürfen eines höheren Aufwands für ihre Sicherung. Der Wert eines Systeminhalts oder Systembestandteils basiert dabei auf dem Schaden, den sein Ausfall, seine unberechtigte Veränderung oder seine Veröffentlichung für den Betreiber der verteilten Krankenakte oder die Systembeteiligten bedeutet. Das Pattern „Threat Assessment“ dient der Bedrohungsanalyse. Im Zuge des Threat Assessments werden konkrete Bedrohungen für die im Rahmen der Anwendung des Patterns „Security needs Identification for Enterprise Assets“ definierten sicherheitsrelevanten Bereiche ermittelt und bewertet. Bei verteilten Krankenakten sind die sicherheitsrelevanten Bereiche primär die verschiedenen Arten von Patienten- und Falldaten sowie die technisch mit der verteilten Akte verbundenen Medizingeräte. Das Pattern „Vulnerability Assessment“ beschreibt die systematische Ermittlung und Katalogisierung von Schwachstellen eines Systems und der organisatorischen Rahmenbedingungen. Es baut wie die drei anderen Patterns der Gruppe „Analyse der Sicherheitssituation“ auf den Ergebnissen des Patterns „Security needs Identification for Enterprise Assets“ auf und stellt seine Ergebnisse den Patterns der Gruppe „Bewertung der Risikosituation“ als eingehende Information zur Verfügung.

Die drei Patterns „Threat Assessment“, „Vulnerability Assessment“ und „Asset Valuation” verweisen mit ausgehenden Führt-zu-Beziehungen auf die Gruppe „Bewertung der Risikosituation“ (vgl. Abbildung 57). Sie liefern somit die benötigte Ausgangsinformation für die Anwendung des Patterns „Risk Determination“. Es ist das einzige, in der Gruppe „Bewertung der Risikosituation“ enthaltene Pattern.

Pattern der Gruppe „Bewertung der Risikosituation“:

- Risk Determination (Anhang S. 525).

Das Pattern „Risk Determination“ beschreibt die systematische Ermittlung des vorliegenden Sicherheitsrisikos für jeden der in den vorherigen Analyseschritten ermittelten Teilbereiche des Systems verteilte Krankenakte und dessen Umgebung.

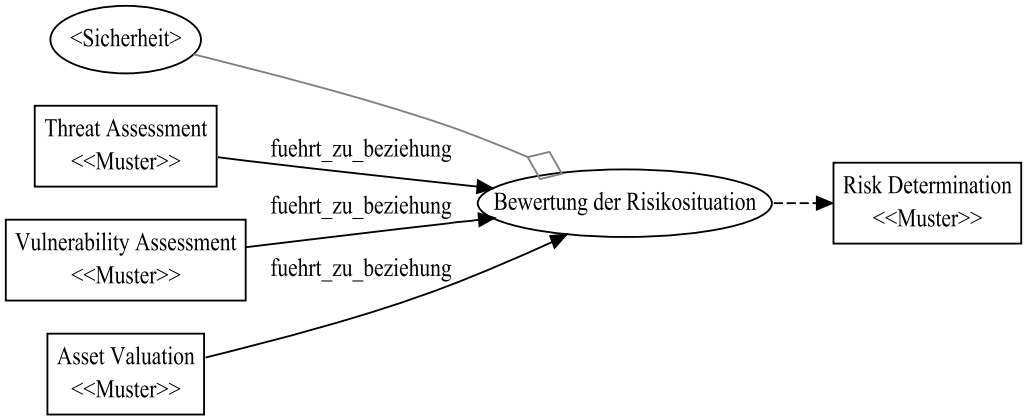



Abbildung 57: Direkte eingehende Abhängigkeiten der Gruppe „Bewertung der Risikosituation“

Bei der Anwendung des Risk-Determination-Patterns werden die Ergebnisse der Patterns „Threat Assessment“ und „Vulnerability Assessment“ (siehe Abbildung 57) nochmals detailliert betrachtet und in Form einer Threat-vulnerability-Table, also einer Bedrohungs- und Verwundbarkeitstabelle zusammengefasst (Schumacher u. a. 2005:S. 138). Die Inhalte dieser Tabelle sind mit den Ergebnissen aus der Anwendung des Asset-Valuation-Patterns, einer Asset-valuation-Table (vgl. Schumacher u. a. 2005:S. 139), in Beziehung gesetzt. Jeder der so konkretisierten Bedrohungen wird zudem eine Wahrscheinlichkeitsbewertung und jeder Verwundbarkeit eine Bewertung des zu erwartenden Schweregrads zugeordnet (Schumacher u. a. 2005:S. 138).

Das Ergebnis dieses Vorgehens sind katalogisierte und bewertete Sicherheitsrisiken für alle relevanten Bereiche einer verteilten Krankenakte. Die bewerteten Sicherheitsrisiken geben Aufschluss über die für das konkrete zu realisierenden System zu erwartenden Sicherheitsprobleme und sind so die Basis für die Entwicklung entsprechender Lösungsansätze. Aus diesem Grund besteht die weitere sicherheitsbezogene Anforderungsanalyse aus den Patterns der Gruppe „Analyse von Sicherheitsproblemen und Lösungsansätzen“.

Patterns der Gruppe „Analyse von Sicherheitsproblemen und Lösungsansätzen“:

- Enterprise Partner Communication (Anhang S. 355)
- Enterprise Security Approaches (Anhang S. 356)
- Enterprise Security Services (Anhang S. 357).

Die Patterns dieser Gruppe dienen speziell der kommunikationsorientierten Betrachtung von Sicherheitsproblemen und geeigneten Lösungen. Das Pattern „Enterprise Partner Communication“ dient dabei als eines der Einstiegspatterns in die detaillierte Untersuchung (vgl. Abbildung 56). Es baut direkt auf den Ergebnissen des Patterns „Risk Determination“ auf und dient der Untersuchung der Kommunikationsbeziehungen zwischen kooperierenden Einrichtungen oder Unternehmen. Bezogen auf die Anwendung bei der Entwicklung verteilter Krankenakten dient das Pattern der Betrachtung der Kommunikationsbeziehungen zwischen

kooperierenden medizinischen Einrichtungen. Die vollständige Anwendung dieses Patterns setzt eine durch die Patterns des Schwerpunkts *Kommunikation* bereits weitgehend definierte Kommunikationsarchitektur voraus. Abhängig von der gewählten Art der Kommunikation gestaltet sich die konkrete Umsetzung des Patterns unterschiedlich (vgl. Schumacher u. a. 2005:S. 180–181). Das Ergebnis der Anwendung des Patterns ist eine Sammlung von Anforderungen an die Sicherung der Kommunikationswege. Diese Anforderungen gehen auch als Ausgangsinformation in die Anwendung der Patterns der Gruppe „Ermittlung der Sicherheitsanforderungen" ein. Sie bilden dort die Basis für die Definition von noch konkreter lösungsbezogenen Anforderungen.

Auch das Pattern „Enterprise Security Approaches" baut direkt auf den Ergebnissen der Risk Determination auf. Es dient der Festlegung, welche Sicherheitsmechanismen in welchen Bereichen des Systems angewandt werden sollen. Unter Sicherheitsmechanismen werden hier Mechanismen wie die Einschränkung des Zugriffs, die Überwachung bestimmter Eigenschaften oder definierte Reaktionen auf erkannte Sicherheitsverletzungen verstanden (vgl. Schumacher u. a. 2005:S. 148 unten). Das „Pattern Enterprise Security Approaches" führt die ermittelten Risiken mit Maßnahmen bzw. Lösungsansätzen zur Erreichung adäquater Sicherheit zusammen. Das Ergebnis wird (vgl. Schumacher u. a. 2005:S. 159) in einer Tabelle mit den Spalten *Properties and Applicability*, *Security Approach*, *Priority* und einer zusätzlichen Spalte für Notizen und Anmerkungen dargestellt.

Diese Tabelle ist, wie in Abbildung 56 durch eine Führt-zu-Beziehung vermerkt, die Ausgangssituation für die Anwendung des Patterns „Enterprise Security Services". Es verfeinert die bisher ermittelten Ergebnisse. Das führt zu einer Auflistung der zu schützenden Bereiche der Anlagen und Daten mit ihren zu schützenden Eigenschaft, dem zu nutzenden Sicherheitsmechanismus[17] und deren Priorität. Des Weiteren werden den so ermittelten Tupeln konkrete Security Services zugeordnet. Beispiel für solche Security Services sind Authentifizierungsdienste (Identification and Authentication) oder Autorisierungsdienste (Access Control) (vgl. Schumacher u. a. 2005:S. 165, Tabelle 6.31). Die Eigenschaften dieser Security Services können durch die Patterns der Gruppe „Ermittlung der Sicherheitsanforderungen" als einzelne umsetzbare Anforderungen an das Softwaresystem einer verteilten Krankenakte definiert werden.

Die Gruppe „Ermittlung der Sicherheitsanforderungen" baut auf den Ergebnissen des Patterns „Enterprise Security Services" auf. Daneben bedürfen seine Patterns, wie in Abbildung 58 dargestellt, noch der Ergebnisse verschiedener anderer Patterns. Die Gruppe selbst enthält acht Patterns, die zu unterschiedlichen sicherheitsrelevanten Eigenschaften von Softwaresystemen Leitlinien für die Definition konkreter Anforderungen enthalten.

Patterns der Gruppe „Ermittlung der Sicherheitsanforderungen":

- Access Control Requirements (Anhang S. 248)
- Audit Requirements (Anhang S. 282)
- Audit Trails and Logging Requirements (Anhang S. 284)

[17] Prevention, Detection und Response (vgl. Schumacher u. a. 2005:S. 171)

- I&A Requirements (Anhang S. 400)
- Intrusion Detection Requirements (Anhang S. 415)
- Non-Repudiation Requirements (Anhang S. 457)
- Roles (Anhang S. 529)
- Security Accounting Requirements (Anhang S. 548).

Wie die Abbildung 58 zeigt, ist die Anforderungsdefinition unter Zuhilfenahme der Patterns der Gruppe „Ermittlung der Sicherheitsanforderungen“ abhängig von den Ergebnissen der Patterns „Risk Determination“, „Enterprise Security Services“, „Enterprise Partner Communication“ und „Analyse der Organisationsstruktur“. Diese Abhängigkeiten sind je nach gewähltem Pattern der Gruppe unterschiedlich intensiv.

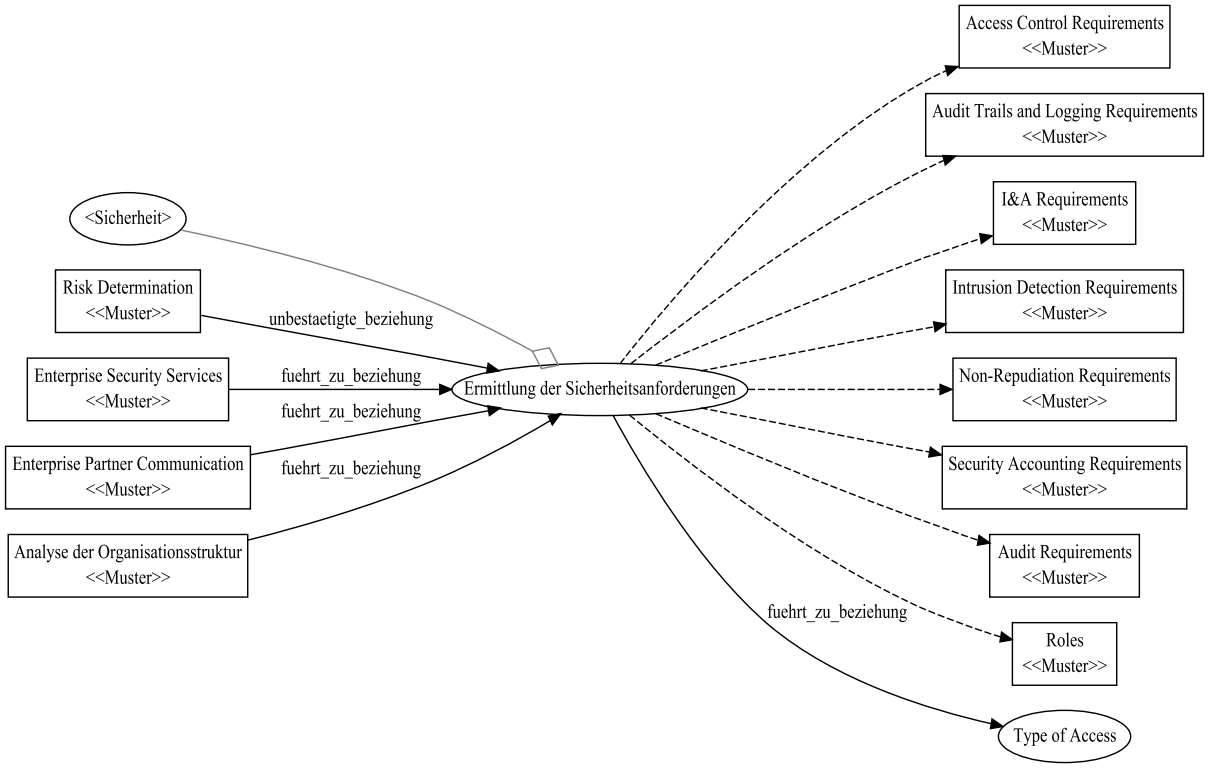



Abbildung 58: Abhängigkeiten der Gruppe „Ermittlung der Sicherheitsanforderungen“

So ist beispielsweise bei der Anwendung des Roles-Patterns eine besondere Abhängigkeit von den Ergebnissen der Analyse der Organisationsstruktur zu identifizieren, da die Organisations- und Aufgabenstruktur rund um das System verteilte Krankenakte für die Definition von Sicherheitsrollen wesentliche Informationen enthält. Aus der Definition der Rollen mittels des Roles-Patterns lassen sich während der Planung von Architektur und Design sowie später bei der Umsetzung konkrete Rechte extrahieren, die zur technischen Umsetzung der abstrakt formulierten Rollen benötigt werden. Die Definition von Rollen als Anforderungen an das Softwaresystem ist eine der Grundvoraussetzungen für ein problemadäquates Berechtigungswesen.

Zwei weitere Patterns, die unter Zuhilfenahme von Ergebnissen anderer Patterns direkt auf die Lösung offener Fragen aus der Anwendung des Patterns „Enterprise Security Services“ eingehen, sind die Patterns „I&A Requirements“ und „Access Control Requirements“. Sie erläutern erprobte Verfahren zur Definition der Anforderungen an die Authentifizierung bzw.

Authentisierung von Benutzern und die Einschränkung von Zugriffsrechten auf Systembestandteile. Eine zuverlässige Authentifizierung und zuverlässige Verfahren zur Einschränkung von Zugriffsrechten sind bei verteilten Krankenakten aufgrund der hohen Anforderungen an den Datenschutz (vgl. Kapitel 2.4.2, S. 63) von besonderer Relevanz.

Neben dem Datenschutz ist auch die Verlässlichkeit der Daten und die zuverlässige Nachvollziehbarkeit möglicher Manipulationen eine zentrale Anforderung an medizinische Informationssysteme (vgl. Kapitel 2.4.2, S. 63). Das Pattern „Non-Repudiation Requirements“ beschreibt ein erprobtes Verfahren zur Definition von Anforderungen an Maßnahmen zur nicht abstreitbaren Dokumentation von Systemvorgängen. Das Pattern dient dabei der Intensivierung bzw. Ergänzung von Maßnahmen, die teils bereits durch die Anwendung des Patterns „Audit Requirements“ als Bestandteil der Systemanforderungen zu dokumentieren sind.

Durch die Anwendung der Patterns aus der Gruppe „Ermittlung der Sicherheitsanforderungen“ als letzte Stufe der Phase *Anforderungsanalyse* im Schwerpunkt *Sicherheit* erfolgt eine detaillierte Dokumentation einzelner Anforderungen an die Sicherheitsarchitektur des Softwaresystems einer verteilten Krankenakte. Die resultierenden Anforderungen sind aufgrund der Stufenweise verfeinernden Analyse im Rahmen der Phase *Anforderungsanalyse* ausreichend feingranular um direkt als Basis für die Auswahl geeigneter Architektur- und Design-Patterns für die Sicherheit der verteilten Krankenakte verwendet zu werden.

### 5.3.2. Architektur

Die Patterns der Ebene *Architektur* im Schwerpunkt *Sicherheit* beschreiben die Gestaltung des Softwaresystems einer verteilten Krankenakte und der umgebenden Infrastruktur (z. B. der Netzwerkinfrastruktur) nach verschiedenen Aspekten der Sicherheit. Die verschiedenen enthaltenen Pattern-Gruppen berücksichtigen für ihren thematischen Bereich jeweils unterschiedliche Lösungsansätze. Die thematischen Bereiche der Gruppen umfassen primär das Softwaresystem selbst, aber in ausgewählten Bereichen auch seine technische Umgebung.

Gruppen der Ebene *Architektur* im Schwerpunkt *Sicherheit*:

- Zugriffbeschränkung (Access Restriction)
- Verteilung der Autorisierungsinformation
- Krankenaktenspezifische Berechtigungskonzepte
- Anzahl der Authentifizierungskriterien
- Anzahl der Authentifizierungsquellen
- Grundlegende Authentifizierungskonzepte
- Identitätsmanagement
- Gestaltung des Zugriffspunkts (Point of Access)
- Sicherheitsbezogenes Logging
- Typ der Zugriffsbeschränkung (Type of Access).

Die Ebene *Architektur* im Schwerpunkt *Sicherheit* enthält die aufgelisteten zehn Pattern-Gruppen. Sie beschreiben Lösungsansätze zur Sicherung des Systems mittels Restriktion des Zugriffs auf die zugrunde liegende technische Kommunikationsinfrastruktur, spezielle Lösungen zur Verbesserung der Sicherheit von verteilten Krankenakten und allgemeine oder grundlegende Konzepte zur Verbesserung bzw. Gewährleistung von Sicherheit in Softwaresystemen.

Der Bereich der technischen Umgebung und Infrastruktur wird primär durch die Gruppe „Access Restriction“ bedient. Sie beinhaltet eine Sammlung von Patterns zur Verbesserung der netzwerkseitigen Sicherheit.

Gruppe „Access Restriction”:

- Demilitarized Zone (Anhang S. 344)
- Front Door (Anhang S. 391)
- Packet Filter Firewall (Anhang S. 464)
- Protection Reverse Proxy (Anhang S. 481)
- Proxy based Firewall (Anhang S. 487)
- Reverse Proxy (Anhang S. 520)
- Stateful Firewall (Anhang S. 579).

Die Patterns dieser Gruppe reichen von Lösungsansätzen, die ausschließlich auf der Kombination und Konfiguration verschiedener Netzwerkkomponenten und deren enthaltener Software basieren (wie z. B. das Pattern Demilitarized Zone), bis hin zu Patterns, deren Bestandteile im Rahmen ihrer Sicherungsaufgaben bereits in eingeschränktem Umfang mit dem Softwaresystem der verteilten Krankenakte in Kommunikation stehen (wie z. B. das Pattern „Front Door"). So kann etwa bei der Anwendung des Patterns „Front Door" in Kombination mit einem oder mehreren der Reverse-Proxy-Patterns (Anhang S. 408 u. 481) die Weitergabe von Information über den in der aktuellen Sitzung (vgl. Pattern „Security Session" S. 553) authentifizierten Benutzer an die eigentliche Anwendungssoftware benötigt werden (vgl. Schumacher u. a. 2005:S. 473 Abschnitt: Context).

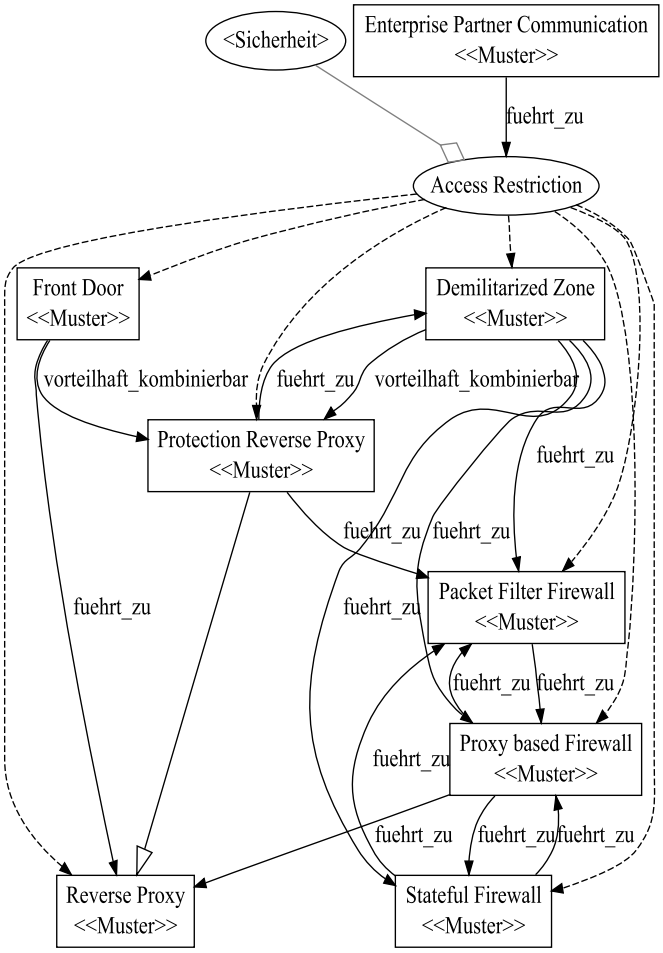



Abbildung 59: Interne Beziehungen der Gruppe „Access Restriction"

Eingehende Anforderungen erhalten die Patterns der Gruppe gemeinsam aus den Ergebnissen der Anwendung des Patterns „Enterprise Partner Communication" sowie durch jeweils individuelle Abhängigkeiten zwischen den enthaltenen Patterns und einzelnen Patterns der Ebene *Anforderungsanalyse*.

Wie die Abbildung 59 zeigt, ist innerhalb der Gruppe „Access Restriction" eine große Vielfalt an Beziehungen verfügbar. Diese Beziehungen, die alle vom Typ „führt zu" sind, weisen auf eine gute Kombinierbarkeit der in der Gruppe enthaltenen Patterns hin. So führt beispielsweise die in „Security Patterns" (Schumacher u. a. 2005:S. 475 ff.) beschriebene Anwendung des Front-Door-Patterns zu einer gleichzeitigen, einander ergänzenden Nutzung

mit den Patterns „Demilitarized Zone“, „Integration Reverse Proxy“ und „Protection Reverse Proxy“. Sie führt somit indirekt zur Nutzung von Patterns der Gruppen „Authentication“ und „Access Control or Authorization“. Durch die Anwendung des Patterns „Demilitarized Zone“ führt „Front Door“, ebenfalls indirekt, zur Anwendung eines oder mehrerer Firewall-Patterns, vorzugsweise des Packet-Filter-Firewall-Patterns.

Das mehrfach erwähnte Pattern „Integration Reverse Proxy“, das ebenfalls eine Spezialisierungsform des Patterns „Reverse Proxy“ ist, ist nicht Bestandteil des Schwerpunkts *Sicherheit*, sondern dem Schwerpunkt *Flexibilität* zugeordnet. Diese Zuordnung ist damit zu begründen, dass die primäre Auswirkung des Integration Reverse Proxy ist, eine größere Zahl von Dienstanbietern flexibel in einen gemeinsamen Zugriffspunkt zu integrieren. Dadurch ermöglicht es z. B. den flexiblen Austausch oder das flexible Verschieben von Diensten auf andere Hardware, ohne dass deren Zugriffsweg und Authentifizierungsmechanismus sich ändern muss.

Neben der Beschränkung des Zugriffs auf das Gesamtsystem durch infrastrukturelle und sonstige technische Maßnahmen ist bei verteilten Krankenakten auch die Beschränkung des Zugriffs und der einzelnen Aktionen des Zugriffs auf die feingranular gegliederten Bestandteile der Akten relevant (vgl. Kapitel 2.4.2). Die dazu erforderlichen Regeln sind sehr komplex und können auf unterschiedliche Weise verwaltet werden. Insbesondere bei verteilten Krankenakten, die eine Menge von Quellsystemen mit häufig hoher Heterogenität beinhalten, kann die Verwendung einer geeigneten Art der Verwaltung von Autorisierungsregeln ein zentraler Aspekt für die dauerhafte Einsatzfähigkeit des Systems sein.

Gruppe „Verteilung der Autorisierungsinformation“:

- Lokale Autorisierungsregeln (Anhang S. 427)
- Zentrale Autorisierungsregeln (Anhang S. 608).

Grundsätzlich lässt sich die Verteilung der Autorisierungsinformation in zwei grundlegende Typen, dargestellt als zwei Patterns, gliedern. Sie beschreiben die Entwicklung von Akten unter Verwendung der lokalen Autorisierungsregeln der einzelnen Quellsysteme ebenso wie Akten mit zentral verwalteten, systemweit geltenden Autorisierungsregeln.

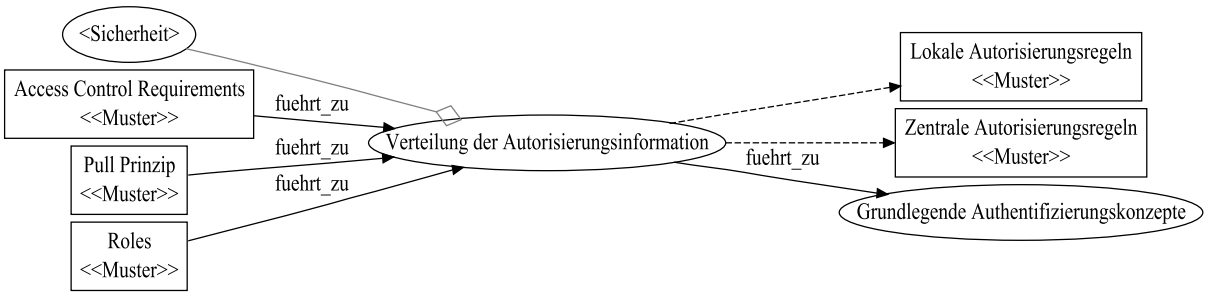



Abbildung 60: Die Gruppe „Verteilung der Autorisierungsinformation“ mit direkten eingehenden Abhängigkeiten

Bei der Verwendung von lokal in den Quellsystemen verwalteten Autorisierungsregeln kann beim Aufbau einer verteilten Krankenakte auf bereits in den Quellsystemen existierende Komponenten zur Verwaltung von Berechtigungen zurückgegriffen werden, sofern die

verteilte Krankenakte auf der Basis bereits bestehender Krankenhausinformationssysteme aufgebaut wird. Die Nutzung bestehender Komponenten führt dazu, dass für das Verwalten und Anwenden von Benutzerrechten keine Veränderung der Software in den Quellsystemen notwendig ist. Es führt allerdings gleichzeitig zu dem Nachteil, dass teilweise nicht für eine einrichtungsübergreifende Nutzung konzipierte Berechtigungssysteme artfremd in der komplexeren Umgebung einer verteilten Krankenakte verwendet werden. Daraus kann ein erhöhter Aufwand für die Pflege der meist für einzelne Personen definierten Regeln in einer Vielzahl verschiedener Quellsysteme resultieren. Diesem Nachteil kann durch die Entwicklung einer Lösung mit zentral verwalteten Autorisierungsregeln begegnet werden. Ein Beispiel zur Verwendung zentral verwalteter Autorisierungsregeln liefert Bergmann (Bergmann 2006:S. 56, vorletzter Absatz) in seinen Aussagen zur *„Infrastrukturbasierten Authentifikation und Autorisierung“* mit der Zuweisung struktureller Rollen im Rahmen des elektronischen Heilberufsausweises. Trotz der genannten Vorteile haftet der Verwendung des Patterns „Zentral verwaltete Autorisierungsregeln“ der Makel an, dass für seine Verwendung alle Quellsysteme einer verteilten Krankenakte sowie die Systeme der verteilten Krankenakte selbst technisch in der Lage sein müssen, eine gemeinsame zentrale Berechtigungsinfrastruktur zu nutzen.

Welchen Kriterien die Berechtigungsregeln, die in einem solchen zentralen oder auch dezentralen System verwaltet werden, selbst genügen müssen, beschreibt Haas in (Haas 2009) auf den Seiten 490 ff. detailliert. Der in dieser Quelle vorherrschende Grad an Detailliertheit ist für eine Anwendung im Rahmen der Entwicklung eines Software-Architekturkonzepts zu fein. Drei grundlegende Konzepte der Berechtigung in verteilten Krankenakten, die schon zum Zeitpunkt der Architekturauswahl berücksichtigt werden können, sind die Patterns der Gruppe „Krankenaktenspezifische Berechtigungskonzepte“.

Pattern der Gruppe „Krankenaktenspezifische Berechtigungskonzepte“:

- Case based Authorization (Anhang S. 305)
- File Authorization (Anhang S. 381)
- Patient based Access Control (Anhang S. 467).

Abhängig von den im Verbreitungsgebiet der verteilten Krankenakte vorliegenden gesetzlichen Rahmenbedingungen sind nicht immer alle der drei grundlegenden Konzepte ohne Kombination mit einem anderen Konzept der Gruppe bzw. mit einem anderweitigen Mechanismus zulässig.

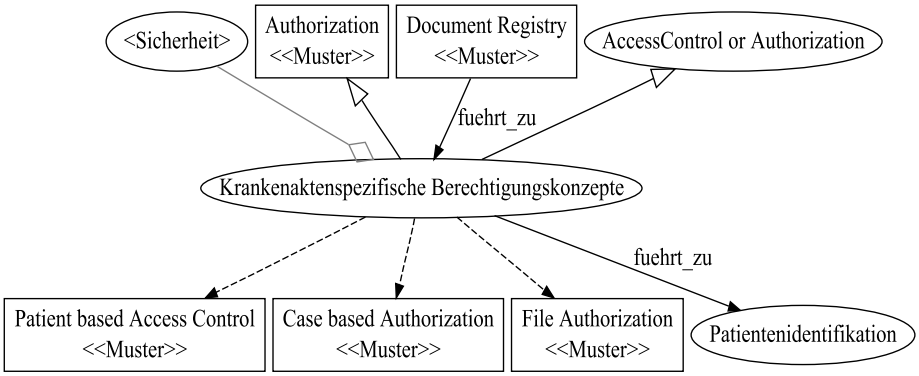



Abbildung 61: Gruppe „Krankenaktenspezifischer Berechtigungskonzepte“ mit eingehenden Abhängigkeiten

Das Pattern „Patient based Access Control“ regelt die Steuerung des Zugriffs über Berechtigungen auf die jeweils vollständige Akte eines Patienten. Das bedeutet, dass Ärzte und medizinisches Personal im Rahmen der Erbringung von Leistungen an einem Patienten auf dessen gesamte Krankenakte zugreifen können.

Entsprechend der in den Quellen (C. Dierks 1999) und (UAG KIS 2011) enthaltenen Darstellungen zu den datenschutzrechtlichen Rahmenbedingungen in Deutschland ist davon auszugehen, dass die alleinige Nutzung des Patterns „Patient based Access Control“ ohne Kombination mit weiteren einschränkenden Mechanismen nicht ausreichend ist, um die in Deutschland geltenden Anforderungen des Datenschutzes zu erfüllen. Das Pattern „Patient based Access Control“ kann aber z. B. in einem System, das in einem Land mit schlechter medizinischer Versorgungsinfrastruktur und niedrigen Datenschutzstandards (z. B. in Entwicklungsländern) eingesetzt wird, auch allein verwendet ein ausreichendes Berechtigungskonzept sein.

In der „Orientierungshilfe Krankenhausinformationssysteme“ (UAG KIS 2011:Teil1, Normative Eckpunkte) wird detailliert beschrieben, wie der Zugriff auf bestehende medizinische Daten von der Aufnahme über die Behandlung bis hin zur Entlassung des Patienten nach geltenden deutschen Datenschutzbestimmungen zu geschehen hat. Diese komplexen Anforderungen setzten eine Kombination von zwei oder drei Patterns der Gruppe „Krankenaktenspezifische Berechtigungskonzepte“ mit dem Access-Control-List-Pattern (Anhang S. 246) voraus.

Grundsätzlich erfordert die Verwendung der Patterns der Gruppe „Krankenaktenspezifische Berechtigungskonzepte“ die eindeutige Identifizierbarkeit der Patienten und bei verteilten Krankenakten insbesondere die eindeutige Identifizierbarkeit der Patienten über Subsystemgrenzen hinweg. Diese Abhängigkeit ist in Abbildung 61 durch die Führt-zu-Beziehung zur Gruppe „Patientenidentifikation“ dargestellt.

Verteilte Krankenakten sind Softwaresysteme. Aus diesem Grund können auch viele allgemeine Sicherheits-Patterns, die für beliebige Softwaresysteme konzipiert sind, genutzt werden, um die Sicherheit von elektronischen und verteilten Krankenakten zu verbessern. Eine wichtige Grundvoraussetzung für jedes durch Rechte, Rollen und Einzelberechtigungen geschützte System ist die zuverlässige und sichere Identifikation des mit dem System interagierenden Subjekts. Die dazu notwendigen Patterns sind in der Gruppe der grundlegenden Authentifizierungskonzepte zusammengefasst.

Inhalt der Gruppe „Grundlegende Authentifizierungskonzepte“:

Enthaltenes Pattern:

- Automated I&A Design Alternatives (Anhang S. 291).

Untergeordnete Gruppen:

- Anzahl der Authentifizierungskriterien (siehe S. 177)
- Anzahl der Authentifizierungsquellen (siehe S. 178).

Die Gruppe enthält das Pattern „Automated I&A Design Alternatives“ sowie die beiden untergeordneten Gruppen „Anzahl der Authentifizierungskriterien“ und „Anzahl der Authentifizierungsquellen“. Sie gliedern die enthaltenen Authentifizierungspatterns nach der Anzahl der Authentifizierungskriterien und der Anzahl der Authentifizierungsquellen. Die Abbildung 62 zeigt die Gruppe einschließlich ihres enthaltenen Patterns und der enthaltenen untergeordneten Gruppen mit den zugehörigen eingehenden Abhängigkeiten. Dabei wird deutlich, dass die Auswahl geeigneter Authentifizierungskonzepte hauptsächlich von den Ergebnissen der Patterns „Non-Repudiation Requirements“ und „I&A Requirements“ abhängig ist.

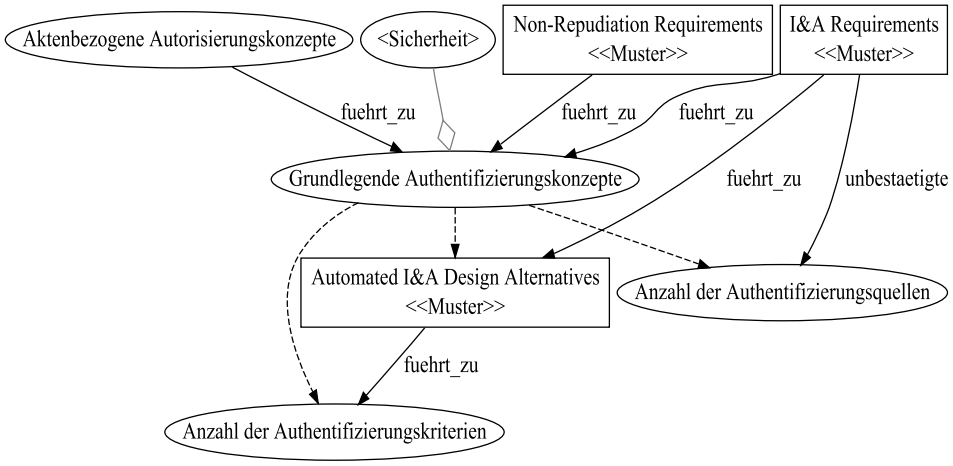



Abbildung 62: Gruppe „Grundlegende Authentifizierungskonzepte“ mit Abhängigkeiten

Das Pattern „Automated I&A Requirements“ dient der Unterstützung bei der Auswahl einer problemadäquaten Authentifizierungsstrategie. Hierbei wird die Problemangemessenheit durch eine priorisierte Betrachtung der I&A Requirements sowie der Berücksichtigung allgemeiner Sicherheitsanforderungen an verteilte Krankenakten erreicht. Die Auswahl einer Authentifizierungsstrategie kann sowohl die Auswahl eines einzelnen Authentifizierungsverfahrens, mehrerer alternativ zueinander verwendbarer Verfahren als auch die Verwendung eines kombinierten Verfahrens bedeuten (vgl. Schumacher u. a. 2005:S. 208). Dadurch hat das Ergebnis des Patterns „Automated I&A Requirements“ einen unmittelbaren Einfluss auf die Auswahl innerhalb der Gruppe „Anzahl der Authentifizierungskriterien“.

Gruppe „Anzahl der Authentifizierungskriterien“:

- Einfaktorige Authentifizierung (Anhang S. 354)
- Mehrfaktorige Authentifizierung (Anhang S. 437).

Die Gruppe „Anzahl der Authentifizierungskriterien“ umfasst die zwei Patterns „Einfaktorige Authentifizierung“ und „Mehrfaktorige Authentifizierung“. Das Pattern „Mehrfaktorige Authentifizierung“ beschreibt dabei die kombinierten Verfahren. Typische Vertreter kombinierter Verfahren sind hierbei die Kombination aus Benutzername, Passwort oder PIN und einem Hardwaretoken oder die Kombination aus Benutzername, Passwort und Zertifikaten einer PKI[18] als sogenannte Mehr-Faktor-Authentifikation (Eckert 2011:S. 431).

[18] PKI = Public Key Infrastructure

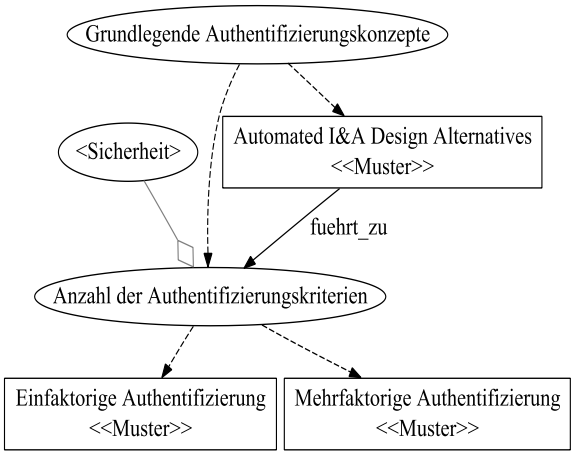



Abbildung 63: Eingehende Abhängigkeiten der Gruppe „Anzahl der Authentifizierungskriterien“

Das Pattern „Einfaktorige Authentifizierung“ beschreibt hingegen die Verwendung von einem oder mehreren nicht kombinierten Authentifizierungsverfahren. Das bedeutet, das Pattern fasst alle Authentifizierungsstrategien zusammen, die lediglich auf einem im Wissen oder Besitz des sich authentisierenden Subjekts befindlichen Faktor oder Kriterium basieren.

Verteilte Krankenakten sind häufig als einrichtungsübergreifende Informationssysteme konzipiert und umgesetzt. Damit ist die Entscheidung über die Architektur einer geeigneten Benutzerverwaltungskomponente, die der verteilten Krankenakte und bzw. oder seinen Subsystemen als Quelle für die zur Authentifizierung von Benutzern benötigten Daten dienen kann, zwangsläufig. Die Gruppe „Anzahl der Authentifizierungsquellen“ fasst verschiedene Patterns zusammen, die unterschiedliche Konzepte zur Speicherung und Integration der Benutzerdaten in zentralen oder dezentral verteilten Benutzerverzeichnissen beschreiben.

Gruppe „Anzahl der Authentifizierungsquellen“:

- Benutzerverwaltung mit dezentralen Daten (Anhang S. 295)
- Meta-Directory (Anhang S. 443)
- Zentrale Benutzerverwaltung (Anhang S. 610).

Die beiden grundlegenden Konzepte innerhalb dieser Gruppe sind in den Patterns „Benutzerverwaltung mit dezentralen Daten“ und „Zentrale Benutzerverwaltung“ beschrieben. Sie bilden die beiden grundsätzlich existierenden Möglichkeiten, Benutzerkonten (einschließlich z. B. der benötigten Passwörter) entweder dezentral in jedem beteiligten Subsystem bzw. in jeder beteiligten Einrichtung für deren Subsysteme oder auch zentral für alle Beteiligten einer verteilten Krankenakte zu speichern und zu verwalten. Das dritte Pattern der Gruppe (Meta-Directory) beschreibt einen Ansatz, unter Verwendung dezentral existierender Benutzerverzeichnisse eine virtuelle Integration vorzunehmen. Dabei wird ein Verzeichnis-Server im Sinne eines „Integration Reverse Proxy“ (siehe Anhang S. 408) zwischen das Softwaresystem der verteilten Krankenakte und die dezentralen Verzeichnisdienste geschaltet. Dieses sogenannte „Meta-Directory“ leitet Authentifizierungsanfragen, Suchanfragen usw. abhängig vom Kontext der Anfrage an den oder die jeweils richtigen dezentralen Verzeichnisdienstserver weiter.

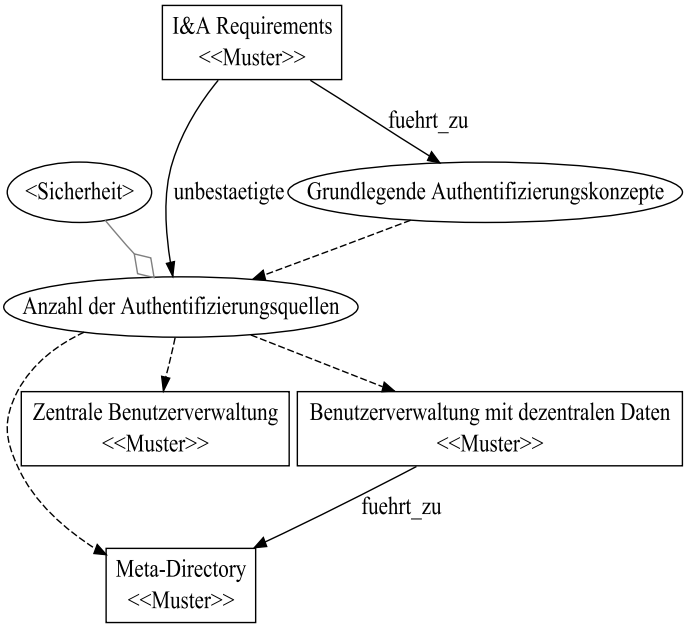



Abbildung 64: Gruppe „Anzahl der Authentifizierungsquellen“ mit internen Abhängigkeiten

Die Abbildung 64 stellt die Abhängigkeiten der Gruppe „Anzahl der Authentifizierungsquellen“ dar. Hier sind im Bereich der eingehenden Beziehungen besonders die Abhängigkeiten von den I&A Requirements sowie von der übergeordneten Gruppe „Grundlegende Authentifizierungskonzepte“ zu erwähnen. Innerhalb der Gruppe „Anzahl der Authentifizierungsquellen“ existiert eine Führt-zu-Beziehung zwischen den Patterns „Benutzerverwaltung mit dezentralen Daten“ und „Meta-Directory“. Diese Beziehung visualisiert, dass die Verwendung eines „Meta-Directorys“ nur dann sinnvoll ist, wenn mindestens zwei oder mehrere Quellen für benutzer- und authentifizierungsrelevante Daten im Gesamtsystem der verteilten Krankenakte existieren.

Die Vielzahl der beteiligten Organisationen und Subsysteme macht die eindeutige Identifizierung von am System beteiligten Subjekten sehr komplex. Besonders komplex wird die Verwaltung von Identitäten dann, wenn in der Gruppe Anzahl der Authentifizierungsquellen nicht das Pattern „Zentrale Benutzerverwaltung“ ausgewählt wird.

Gruppe Identitätsmanagement:

Patterns:

- Circle of Trust (Anhang S. 315)
- Identity Federation (Anhang S. 403)
- Identity Provider (Anhang S. 404)
- Meta-Directory (Anhang S. 443).

Untergeordnete Gruppen:

- Patientenidentifikation (Schwerpunkt *Zuverlässigkeit*).

Unabhängig von der zentralen oder dezentralen Speicherung der Benutzerinformation ist auch die zuverlässige und sichere Weitergabe der Benutzeridentität bei entfernten Aufrufen zwischen den Subsystemen eine komplexe Herausforderung. Um dieser Herausforderung zu

begegnen, sind speziell dafür vorgesehene Patterns in der Gruppe „Identitätsmanagement“ zusammengefasst. Die Gruppe überlappt im Pattern „Meta-Directory“ mit der Gruppe „Anzahl der Authentifizierungsquellen“, es befindet sich im Schnittbereich zwischen beiden Gruppen. Anders als die an der Authentizität des Benutzers orientierten Gruppen befassen sich die Patterns der Gruppe „Identitätsmanagement“ auch mit der Identität von Subjekten, die nicht aktiv am System teilnehmen, sondern nur durch das System verwaltet werden.

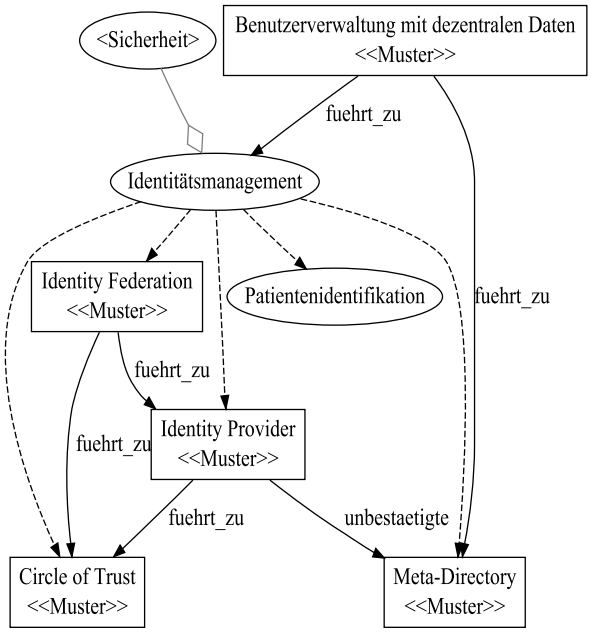



Abbildung 65: Gruppe „Identitätsmanagement“ mit internen Abhängigkeiten

Die Abbildung 65 zeigt die Gruppe „Identitätsmanagement“ mit ihren enthaltenen Patterns und deren gegenseitigen Abhängigkeiten. Die dargestellte Beziehung zwischen dem Pattern „Benutzerverwaltung mit dezentralen Daten“ und der Gruppe „Identitätsmanagement“ zeigt, dass Identitätsmanagement immer dann von besonderer Bedeutung ist, wenn Benutzer, Personen – oder allgemein Subjekte – an mehreren verschiedenen Orten innerhalb eines Gesamtsystems gespeichert und verwaltet werden. Wird in einem System das Pattern „Benutzerverwaltung mit dezentralen Daten“ angewandt, so ist im Rahmen der Konzeption der Architektur auch die Verwendung von Patterns der Gruppe „Identitätsmanagement“ zu prüfen.

Das Pattern „Identity Provider“ beschreibt in abstrakter Form einen Ansatz für die zentrale Verwaltung der Identitäten von beteiligten Subjekten in hoch verteilten und heterogenen Umgebungen (N. Delessy u. a. 2007:S. 2). Ein möglicher Ansatz, eine solche zentrale Verwaltung von Subjekten zu ermöglichen, ist die Verwendung des Meta-Directory-Patterns.

In verteilten Systemen, in denen entfernte Dienstaufrufe über mehrere Schichten hinweg durchgeführt werden, ist die sichere Authentifizierung des aufrufenden Benutzers durch diese Schichten hindurch notwendig. Da prinzipiell der Anbieter eines jeden aufrufenden Dienstes in der Lage ist, Authentifizierungsinformationen zu speichern, widerrechtlich zu verwenden oder zu verfälschen, sind in diesem Bereich weitere sichernde Maßnahmen erforderlich. Das Pattern „Circle of Trust“ beschreibt eine Möglichkeit, die Art und den Grad des Vertrauens

zwischen einer Menge von Serviceanbietern systemseitig abzubilden (N. Delessy u. a. 2007:S. 3).

Das Pattern „Identity Federation“ (N. Delessy u. a. 2007:S. 2 und S. 4) beschreibt, wie eine sichere und zuverlässige Weitergabe der Information über die Identität des aktuellen Akteurs innerhalb einer Kette von verteilten Aufrufen umgesetzt werden kann, die sich nicht auf Dienste innerhalb eines „Circle of Trust“ beschränkt. In seiner Beschreibung weist das Pattern auch auf konkrete Standards zur technologischen Umsetzung, wie z. B. den SAML[19] Standard hin.

Die Gruppe „Identitätsmanagement“ umfasst somit verschiedene Patterns, die helfen, in einem verteilten System mit verschiedenen Orten der Benutzerverwaltung und bzw. oder vielen verschiedenen Diensten, die Information über Benutzer sowie deren Identität bereitzustellen und zuverlässig zur Steuerung von Berechtigungen zu verwenden.

Verteilte Krankenakten werden über die Grenzen von Organisationseinheiten und Einrichtungen hinweg genutzt. Dabei können nicht alle an der Verbindung beteiligten Netze als sicher betrachtet werden. Mit zunehmender Größe eines Gesamtsystems steigt zudem die Wahrscheinlichkeit, dass die Unsicherheit erhöhende Faktoren selbst Bestandteil des Systems sind oder auf Teile der Infrastruktur Einfluss haben können. Öffentlich zugängliche Netze oder die Übergänge zwischen den Netzen von Einrichtungen sind solche Systembestandteile. Für den Zugriff auf Dienste einer verteilten Krankenakte aus einer unsicheren Umgebung heraus ist der Aufbau eines definierten und gut abgesicherten Zugriffswegs notwendig. Er dient sowohl der Bereitstellung der notwendigen Dienste als auch der Abschottung gegenüber unsicheren Umgebungen. Die Gruppe „Point of Access“ beschreibt Patterns zum Aufbau solcher abgesicherten Zugriffswege für schutzbedürftige Dienste.

Gruppe „Point of Access”:

- Checkpoint (Anhang S. 310)
- Secure Access Layer (Anhang S. 547)
- Single Access Point (Anhang S. 566).

Sie umfasst die drei Patterns „Checkpoint“, „Secure Access Layer“ und „Single Access Point“, die allein oder miteinander kombiniert eingesetzt werden können, um einen abgesicherten Zugriff auf eine verteilte Krankenakte oder einzelne Komponenten einer verteilten Krankenakte zu ermöglichen.

Das Pattern „Single Access Point“ beschreibt eine Systemarchitektur, bei der es nur einen Weg gibt, den Dienst einer Komponente oder eines Systems zu erreichen und somit zu nutzen. In Anlehnung an die Patternbeschreibung aus dem Buch „Security Patterns“ (Schumacher u. a. 2005:S. 279 ff.) lässt sich die grundlegende Idee des Patterns wie folgt erklären. In einem Dorf, bestehend aus einer Menge von Häusern mit je einer Tür, müssen für jedes Haus und jede Tür umfangreiche Sicherungsmaßnahmen ergriffen werden. Der

[19] Security Assertion Markup Language, http://docs.oasis-open.org/security/saml/v2.0/

Aufwand ist hoch, da für jedes Haus eigene sichernde Ressourcen vorgehalten werden müssen. Schafft man hingegen eine Mauer, die das Dorf umschließt und selbst nur ein Tor besitzt, so kann ein großer Teil der Sicherungsmaßnahmen für alle Häuser gemeinsam an einer zentralen Stelle und mit reduziertem Bedarf an Ressourcen umgesetzt werden. Gleiches gilt für Softwarekomponenten und Subsysteme. Ein gemeinsamer, für die Komponenten vertrauenswürdiger Zugriffspunkt kann Aufgaben zur Absicherung der Subsysteme und Komponenten gegenüber nicht vertrauenswürdigen Umgebungen zentral übernehmen und so die Komplexität der Implementierung der einzelnen Komponenten reduzieren. Außerdem wird auf diese Weise die Überwachbarkeit des Systems verbessert, weil ein geeigneter zentraler Punkt z. B. für die Durchführung von Logging etc. entsteht.

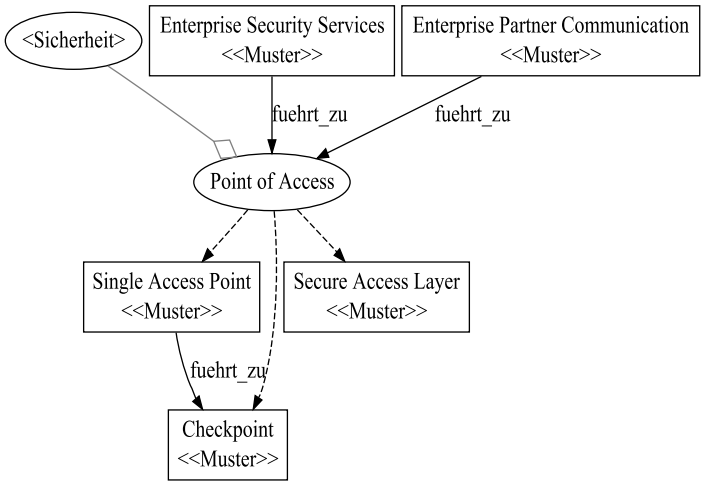



Abbildung 66: Gruppe Point of Access

Das Pattern „Checkpoint“ baut auf dem Pattern „Single Access Point“ auf und erweitert den Zugriffspunkt um die Funktionalität der Authentifizierung oder einer sonstigen Prüfung von Zugangsregeln. Es beschreibt somit die Nutzung eines „Single Access Points“ zum Zweck der Authentifizierung am System. Die Kombination aus beiden Patterns beschreibt somit eine System-Architektur mit einem einzigen möglichen Zugangsweg, bei dessen Nutzung eine Authentifizierung des Benutzers durchgeführt wird. Nicht authentifizierte Nutzer werden abgewiesen und so an der Nutzung der geschützten Dienste gehindert.

Einen etwas anderen Sicherheitsaspekt zeigt das Pattern „Secure Access Layer“. Seine Autoren Yoder und Barclow weisen darauf hin, dass der Ort der Integration zwischen zwei Subsystemen auch dann ein gefährdeter Ort ist, wenn das Single-Access-Point- und das Checkpoint-Pattern eingesetzt werden (Yoder & Barcalow 1998:S. 2). Die Unsicherheit kann dann z. B. in einer nicht bezüglich ihrer Sicherheit optimierten Schnittstelle liegen. Um dieses Defizit zu beheben, empfehlen die beiden Autoren die Verwendung einer oder mehrerer zusätzlicher Schichten (Secure Access Layer) zur Behebung erkennbarer Defizite. Die Gestaltung einer „Secure Access Layer“ kann je nach Anwendungsfall sehr unterschiedlich sein. Sie kann von der Einführung einer zusätzlichen Schicht im Netzwerk-Protokoll-Stack (Yoder & Barcalow 1998:S. 25), z. B. zur verschlüsselten Übertragung, bis hin zu einer zusätzlichen Fassadeschicht reichen, die die angebotenen Operationen der Schnittstelle um zusätzliche Sicherheitsprüfungen erweitert.

Die Abbildung 66 zeigt, dass die konkrete Umsetzung der Patterns der Gruppe „Point of Access“ durch die Ergebnisse aus der Anwendung der Patterns „Enterprise Security Services“ und „Enterprise Partner Communication“ aus der Ebene *Anforderungsanalyse* bestimmt wird.

Eine weitere im Rahmen der Untersuchungen zum Schwerpunkt *Sicherheit* identifizierte Gruppe ist die Gruppe „Sicherheitsbezogenes Logging“. Zu ihr sind im Rahmen dieser Untersuchung keine geeigneten Patterns gefunden worden. Für die Erfüllung der Non-Repudiation-Requirements sind allerdings verlässliche Logging-Mechanismen erforderlich. Zur Vervollständigung einer Pattern-Sprache für verteilte Krankenakten ist es daher notwendig, für die Gruppe „Sicherheitsbezogenes Logging“ geeignete Patterns zu finden oder zu entwickeln.

Während die Patterns der Gruppe „Point of Access“ dazu dienen, den technischen Zugriff auf die verteilte Krankenakte über einen abgesicherten Zugriffsweg zu bündeln und so die Absicherung der Zugriffe vereinfachen, müssen auch innerhalb der Benutzeroberfläche die vom jeweils interagierenden Subjekt abhängigen Berechtigungen durchgesetzt werden. Die grundlegenden Ansätze hierzu sind in der Gruppe „Type of Access“ zusammengefasst.

Gruppe „Type of Access“:

- Full Access with Errors (Anhang S. 393)
- Limited Access (Anhang S. 425).

Innerhalb der Gruppe gibt es zwei verschiedene Ansätze, Zugriffsrestriktionen durchzusetzen und im System zu integrieren. Der Ansatz „Full Access with Errors“ integriert die Berechtigungen nicht in die Art und Weise, wie die Benutzeroberfläche dem Benutzer präsentiert wird, sondern zeigt eine Benutzeroberfläche mit Bedienelementen für alle prinzipiell im System verfügbaren Funktionen und Daten an. Die Zugriffsrestriktionen werden erst im Falle eines Zugriffsversuchs aktiv. Besitzt ein Benutzer keine Berechtigung zum Zugriff auf ein beteiligtes Objekt oder für das Ausführen der gewählten Funktion, so wird ihm das durch eine Fehlermeldung mitgeteilt. Den dazu gegensätzlichen Lösungsansatz verfolgt das Pattern „Limited Access“. Bei Systemen, deren Zugriffsrestriktionen nach dem Limited-Access-Pattern gestaltet sind, werden Objekte und Bedienelemente, zu deren Nutzung ein Benutzer nicht berechtigt ist, nicht angezeigt oder gegebenenfalls als erkennbar inaktiv dargestellt. Auf diese Weise erhält der Benutzer eine seinen Berechtigungen angepasste Oberfläche zur Interaktion mit dem System.

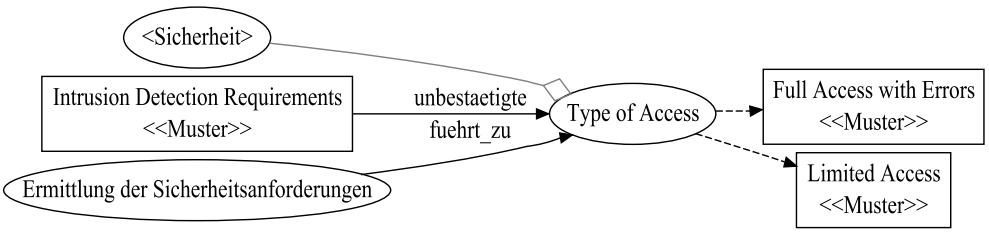



Abbildung 67: Gruppe „Type of Access“ ohne interne Abhängigkeiten

Beide Patterns der Gruppe „Type of Access“ haben unterschiedliche Vor- und Nachteile. Benutzer haben bei der Anwendung des Full-Access-with-Errors-Patterns den Vorteil erkennen zu können, ob sie eine Operation nicht ausführen können, weil sie im System nicht vorgesehen ist oder weil keine Berechtigung dazu vorliegt. Für den Benutzer eines nach dem Limited-Access-Pattern gestalteten Systems ist das nicht durch die Betrachtung der Oberfläche des Systems möglich.

Der zentrale Vorteil des Limited-Access-Patterns ist eine übersichtlichere Benutzeroberfläche. Da sie nur Bedienelemente und Objekte zeigt, zu deren Verwendung der aktuell angemeldete Benutzer berechtigt ist, ist die Menge der verfügbaren Inhalte kleiner und dem Aufgabengebiet des Benutzers angepasst. Dadurch wird die Benutzeroberfläche, bei nach Usability-Kriterien gleichwertiger Gestaltung, übersichtlicher als die Oberfläche eines vergleichbarem Full-Access-with-Errors-Systems (vgl. Schumacher u. a. 2005:S. 313).

### 5.3.3. Design

Für die Umsetzung von Details der Patterns aus der Ebene *Architektur* des Schwerpunkts *Sicherheit* und für den Einbau von speziellen Sicherheitsmechanismen in kleineren Bestandteilen des Softwaresystems einer verteilten Krankenakte sind zusätzlich zu den in den Kapiteln 5.3.1 und 5.3.2 vorgestellten Patterns feingranulare Security-Patterns der Ebene *Design* nötig. Diese Patterns sind in die folgenden Gruppen gegliedert.

Gruppen des Schwerpunkts *Sicherheit* in der Ebene *Design*:

- Access Control or Authorization
- Authentication
- Authentication Data.

Die Gruppe „Access Control or Authorization“ beinhaltet eine Sammlung von Patterns zur Umsetzung von Zugriffsrestriktions- und Autorisierungsmechanismen. Sie beschreiben Mechanismen zur Umsetzung eines Berechtigungswesens auf der Basis von jeweils unterschiedlichen Arten von Quellinformation.

Patterns der Gruppe “Access Control or Authorization”:

- Access Control List (Anhang S. 246)
- Attribute based Access Control (Anhang S. 280)
- Authorization (Anhang S. 288)
- Metadata based Access Control (Anhang S. 445)
- Role based Access Control (Anhang S. 527).

Das Pattern „Access Control List“ (ACL) beschreibt die Berechtigungssteuerung pro Ressource auf Basis von je einer Matrix mit den Achsen *Rechte* und *Subjekte* (Nelly Delessy u. a. 2007:S. 9). Diese Matrix ermöglicht eine direkte Zuordnung zwischen Rechten und Subjekten für je ein zu sicherndes Objekt. Das Role-based-Access-Control-Pattern beschreibt die Gruppierung von Subjekten zu aufgabenbezogenen Rollen. Rollen sind Gruppen von Subjekten, die mindestens in einem fachlichen Aufgabengebiet mit gleichen Berechtigungen

ausgestattet sind. Ein Subjekt, das verschiedene Aufgaben wahrnimmt, kann somit auch in verschiedenen Rollen vertreten sein. Die Patterns „Role based Access Control“ und „Access Control List“ sind gut miteinander kombinierbar. Ein bekanntes Beispiel hierfür ist die Verwaltung der Berechtigungen auf Dateien und Verzeichnisse im Windows-Dateisystem NTFS.

Das Metadata-based-Access-Control-Pattern ermöglicht eine Berechtigungssteuerung auf der Basis von Regeln, die auf die Metadaten des zu schützenden Objekts angewendet werden (siehe Priebe u. a. 2004:S. 10). Metadaten eines Objekts können dabei vom einfachen Erstellungsdatum einer Datei bis hin zu einer Auflistung der postalischen Empfänger eines Arztbriefs reichen. Das Pattern „Attribute based Access Control“ (siehe Priebe u. a. 2005) ist dem ähnlich. Es ermöglicht ebenfalls eine auf Regeln basierende Erteilung von Zugriffsrechten. Die Regeln werden bei diesem Pattern auf die Attribute des zu schützenden Objekts und nicht auf dessen Metadaten angewendet.

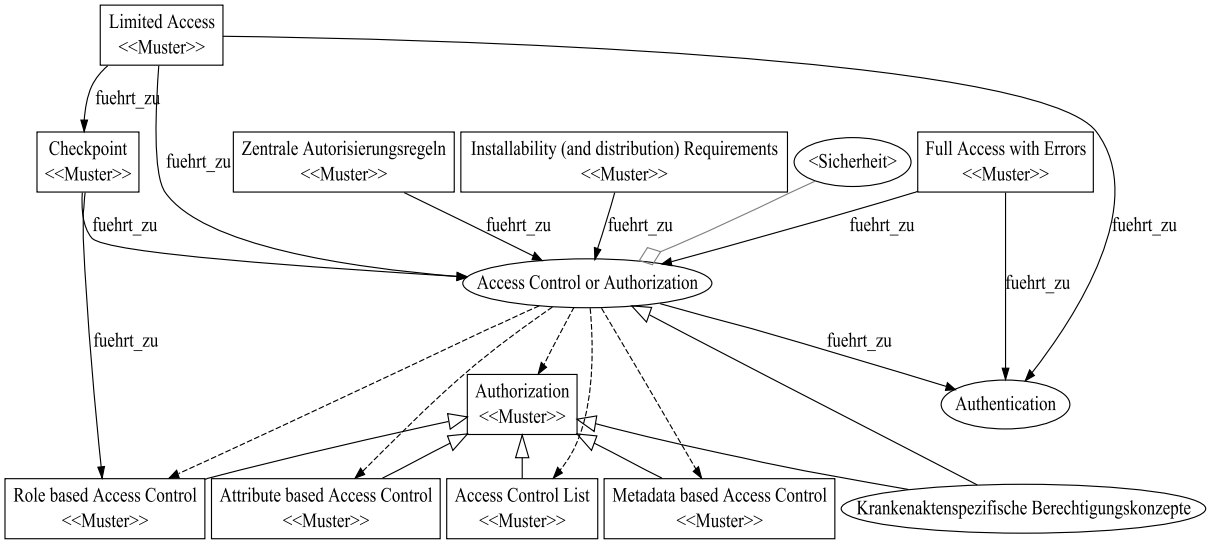


Abbildung 68: Gruppe „Access Control or Authentication“ mit internen Abhängigkeiten

Das Pattern „Authorization“ ist ein sehr abstrakt formuliertes Pattern, das durch verschiedene Patterns konkretisiert wird (Schumacher u. a. 2005:S. 246). Die vier bereits beschriebenen Patterns der Gruppe „Access Control or Authorization“ sind solche Patterns. Die Abbildung 68 zeigt diesen Zusammenhang durch die dargestellten Spezialisierungsbeziehungen. Aus diesem Grund kann das Authorization-Pattern auch als das Pattern bezeichnet werden, das die Gruppe „Access Control or Authorization“ in Form eines abstrakten Patterns beschreibt.

Die Abbildung 68 zeigt außerdem, dass eine Vielzahl der Patterns aus den höheren Ebenen des Schwerpunkts *Sicherheit* direkte Beziehungen zu der Gruppe „Access Control or Authorization“ sowie ihren Patterns besitzt. So erfordert beispielsweise die Umsetzung der Patterns „Limited Access“ und „Full Access with Errors“ zwangsläufig die Verwendung eines Patterns der Gruppe „Access Control or Authorization“. Andere Patterns, wie das Pattern „Zentrale Autorisierungsregeln“ setzen für ihre Anwendung voraus, dass in dem System, in dem sie eingesetzt werden, Autorisierung betrieben wird. Denn in einem System, das ohne Autorisierung auskommt, ist es auch nicht erforderlich Autorisierungsregeln zu verwalten.

Um Benutzer für bestimmten Aktionen autorisieren zu können, ist ihre vorherige Authentifizierung notwendig. In Abbildung 68 wird dieser Zusammenhang durch die Führt-zu-Beziehung zwischen den Gruppen „Access Control or Authorization“ und „Authentication“ abgebildet. Die zugehörige Gruppe „Authentication“ enthält verschiedene Patterns zur konkreten Umsetzung der Benutzerauthentifizierung in Softwaresystemen. Die Gruppe ist dabei nicht als vollständige Sammlung, sondern als laufend zu erweiternde Sammlung typischer Authentifizierungsarten zu verstehen.

Patterns der Gruppe „Authentication“:

- Authenticator (Anhang S. 286)
- Biometrics Design Alternatives (Anhang S. 297)
- Password Design and Use (Anhang S. 465).

Das Pattern „Authenticator“ ist das Pattern, das generell das Problem der Verifikation der Identität des interagierenden Subjekts behandelt (Schumacher u. a. 2005:S. 323). Es wird üblicherweise mit dem Pattern „Single Access Point“ kombiniert (Schumacher u. a. 2005:S. 324; Diagramm siehe Anhang S. 286). Teilweise wird diese Kombinationsbeziehung auch als Spezialisierung des Patterns „Single Access Point“ bezeichnet (z. B. in Schumacher u. a. 2005:S. 327). Zur Umsetzung konkreter Authentifizierungsmechanismen verweist es auf Patterns, die je einen konkreten Mechanismus oder ein konkretes Authentifizierungsprotokoll im Detail beschreiben. Beispiele dafür sind die Patterns „Biometrics Design Alternatives“ oder „Password Design and Use“. Diese Abhängigkeit wird auch in der Abbildung 69 durch die beiden entsprechenden Führt-zu-Beziehungen dargestellt.

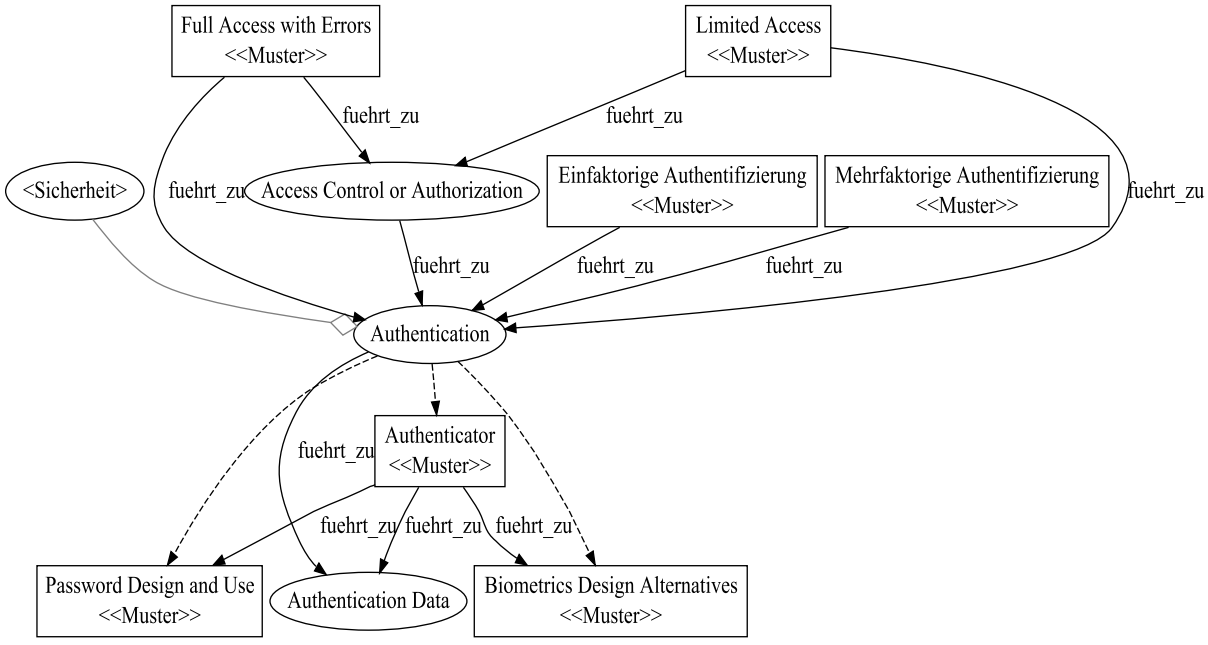



Abbildung 69: Gruppe „Authentication“ mit internen Abhängigkeiten

Die vorherige Auswahl eines der Patterns der Gruppe „Anzahl der Authentifizierungs-kriterien“ (Einfaktorige oder Mehrfaktorige Authentifizierung) bestimmt die Notwendigkeit

der Auswahl eines oder mehrerer Patterns der Gruppe „Authentication“. Sie dienen der konkreten Umsetzung von ausgewählten Ausprägungen des Patterns „I&A Design Alternatives“.

Das Pattern „Password Design and Use“ beschreibt Wege der Gestaltung passwortbasierter Authentifizierungsmechanismen hinsichtlich der Definition von Passwort-Richtlinien, der Nutzung von Passworten und dem Schutz von Passworten (siehe Schumacher u. a. 2005:S. 218–219). Das Biometrics-Design-Alternatives-Pattern beschreibt die Verwendung biometrischer Verfahren zur Authentifizierung von Benutzern (siehe Schumacher u. a. 2005:S. 229 ff.). Weitere Kandidaten für eine Zuordnung zu der Gruppe „Authentication“ sind die Patterns „PKI Design Variables“ und „Hardware Token Design Alternatives“ (vgl. Schumacher u. a. 2005:S. 228).

Die Gruppe „Authentication Data“ befindet sich in einer direkten Abhängigkeit zur Gruppe „Authentication“, insbesondere zu dem enthaltenen Pattern „Authenticator“. Sie beinhaltet in ihrem vorgestellten Zustand nur das Pattern „Credentials“. Über den exakten Namen des Patterns scheinen sich auch seine Autoren nicht eindeutig einig zu sein. Der Titel der Konferenzversion des Papers zur Vorstellung des Patterns lautet „The Credential Pattern“ (Morrison & Eduardo B. Fernandez 2006a), während die in der ACM Digital Library veröffentlichte Version des Papers mit dem Titel „The Credentials Pattern“ (Morrison & Eduardo B. Fernandez 2006b) überschrieben ist. Der häufigeren Verwendung folgend, ist das Pattern in dieser Arbeit unter dem Namen „Credentials“ erfasst.

Gruppe „Authentication Data“:

- Credentials (Anhang S. 332).

Das Credentials-Pattern beschreibt einen flexibel einsetzbaren Ansatz zur Abbildung von zugriffsbezogenen Eigenschaften in Abhängigkeit von einem authentifizierten bzw. authentifizierbaren Subjekt. Die Pattern-Dokumentation (Morrison & Eduardo B. Fernandez 2006a:S. 3–4) beinhaltet drei grundlegende Ausprägungen des Patterns. Die Ausprägung „Issue Credential“ wird als Mittel des anwendungsfallbezogenen Berechtigungsnachweises verwendet. Dabei erhält das authentifizierte Subjekt von einer vertrauenswürdigen Ausgabestelle ein Credential, das zur zuverlässigen Identifizierung des Subjekts bei Drittsystemen verwendet werden kann und zugleich dessen Rollen und Rechte im angeforderten Kontext enthält (vgl. Morrison & Eduardo B. Fernandez 2006a:S. 3). Eine weitere Ausprägung ist die „Principal Authentication“, also die Authentifizierung eines Subjekts anhand eines Credentials. Hier liefert das Subjekt (Principal) ein Credential an einen Authenticator, der durch Prüfung des Credentials die Authentizität des anfragenden Subjekts bestätigt oder verneint. Die einfachste Form dieser Ausprägung ist die Authentifizierung eines Benutzers anhand seiner Kombination aus Benutzername und Passwort, die als Credential an das authentifizierende System übergeben werden. Die dritte Ausprägung, die „Principal Authorization“ beschreibt die Autorisierung eines Subjekts auf Basis eines vorgelegten Credentials.

Typische praktische Vertreter des Credentials-Patterns sind neben Benutzername/Passwort auch Kerberos Tickets, X.509 Zertifikate oder SAML Tokens (Morrison & Eduardo B. Fernandez 2006b:S. 4).

## 5.4. Zuverlässigkeit

Die in medizinischen Informationssystemen verarbeiteten Daten werden verwendet, um die Behandlung eines Patienten zu dokumentieren und im Rahmen dieser Behandlung Entscheidungen zu treffen. Dabei können sowohl fehlerhafte Daten als auch die fehlende Verfügbarkeit von Daten zu Behandlungsfehlern führen und so die Gesundheit oder das Leben des zu Behandelnden gefährden. Aus diesem und anderen Gründen (vgl. Kapitel 2.4.2, insbesondere S. 63) ist die Zuverlässigkeit einer der zentralen Schwerpunkte bei der Entwicklung und Planung verteilter Krankenakten.

Der Aspekt der Zuverlässigkeit umfasst bei verteilten Krankenakten vor allem die folgenden Eigenschaften (vgl. S. 63):

- Verfügbarkeit
- Validität
- Revisionsfähigkeit
- Verbindungsqualität
- Fehlertoleranz
- Skalierbarkeit
- Stabilität.

Die dargestellte Auflistung ist keine Menge mit vollständig disjunkten Elementen. Stattdessen existieren vielfältige Abhängigkeiten zwischen den verschiedenen Eigenschaften. So sind z. B. die Elemente *Verfügbarkeit*, *Verbindungsqualität*, *Fehlertoleranz*, *Skalierbarkeit* und *Stabilität* stark voneinander abhängig. Die vorliegenden Abhängigkeiten sind vielfältig, ihre Art und Wirkungsweise lässt sich allerdings gut anhand eines Beispiels beschreiben. Betrachtet man den Ausfall eines Subsystems als vorliegenden Fehler, so kann durch diesen Fehler die Verfügbarkeit und die Stabilität des Gesamtsystems beeinträchtigt werden. Ist das System allerdings so aufgebaut, dass es gegenüber dieser Art von Fehler tolerant ist und geeignete Kompensationsmechanismen besitzt, so können Stabilität und Verfügbarkeit trotz des genannten Fehlers gewährleistet werden. Das System verhält sich also bezüglich der Ausfälle einzelner Subsystemknoten fehlertolerant.

Auf diese Weise können die Eigenschaften *Verfügbarkeit*, *Verbindungsqualität*, *Skalierbarkeit* und *Stabilität* als vier notwendige Eigenschaften betrachtet werden, deren Beeinträchtigung durch die Toleranz gegenüber Fehlern je einer bestimmten Art vermieden oder kompensiert werden kann. Aus diesem Grund sind im Schwerpunkt *Zuverlässigkeit* besonders viele Patterns aus dem Gebiet der Fehlertoleranz enthalten.

Intern ist der Schwerpunkt *Zuverlässigkeit* wie die drei anderen Schwerpunkte durch die Ebenen *Anforderungsanalyse*, *Architektur* und *Design* strukturiert um eine anwendungs- und granularitätsbezogene Darstellungsform der Patternsammlung zu gewährleisten.

### 5.4.1. Anforderungsanalyse

Im Zuge der Anforderungsanalyse wird das zu planende System auf Einflussfaktoren, die die Zuverlässigkeit des Systems beeinflussen, hin untersucht, um daraus konkrete Anforderungen an die Zuverlässigkeit des Gesamtsystems verteilte Krankenakte abzuleiten. Zu diesem Zweck sind die Patterns der Ebene *Anforderungsanalyse* in drei Gruppen aufgeteilt. Zwei der Gruppen beschreiben die schrittweise Analyse bis hin zu konkreten Anforderungen; die dritte Gruppe beschreibt die Weiterentwicklung der Anforderungen zu späteren Zeitpunkten im Entwicklungs- oder Lebenszyklus der verteilten Krankenakte.

Gruppen:

- Analyse von Faktoren, die die Zuverlässigkeitsanforderungen beeinflussen
- Ermittlung der Zuverlässigkeitsanforderungen
- Überarbeiten der Zuverlässigkeitsanforderungen.

Die Anforderungsanalyse des Schwerpunkts *Zuverlässigkeit* beginnt mit den Patterns der Gruppe „Analyse von Faktoren, die die Zuverlässigkeitsanforderungen beeinflussen“. Sie enthält die beiden Patterns „Analyse der organisatorischen Verteilung im Verbund“ und „Untersuchung betroffener Subsysteme“. Beide Patterns werden auch in der Anforderungsanalyse anderer Schwerpunkte eingesetzt. Ihre Ergebnisse können daher wiederverwendet werden.

Gruppe „Analyse von Faktoren, die die Zuverlässigkeitsanforderungen beeinflussen“:

- Analyse der organisatorischen Verteilung im Verbund (Anhang S. 272)
- Untersuchung betroffener Subsysteme (Anhang S. 595).

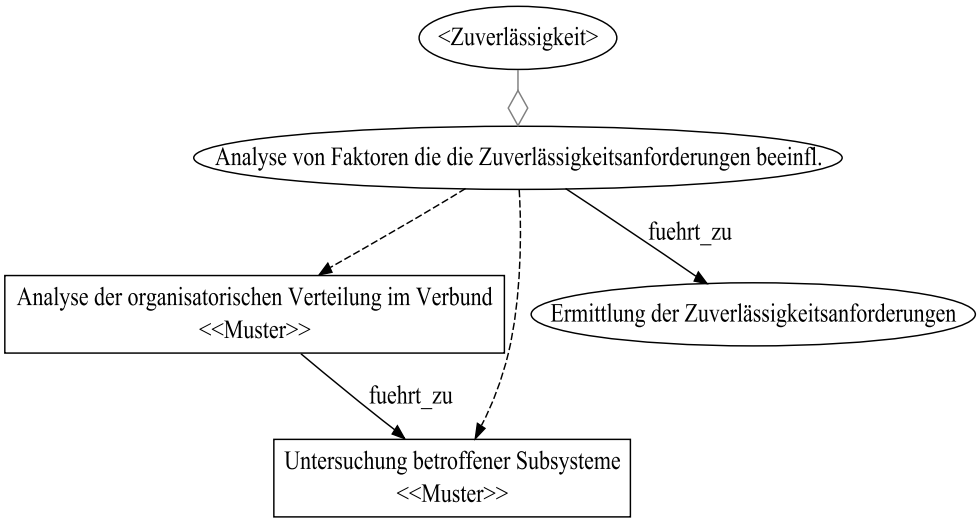



Abbildung 70: Gruppe „Analyse von Faktoren, die die Zuverlässigkeit beeinflussen“ einschließlich interner Abhängigkeiten

Das Pattern „Analyse der organisatorischen Verteilung im Verbund“ findet auch in den Gruppen „Analyse flexibilitätsbeeinflussender Faktoren“ und „Analyse des Kommunikationsbedarfs“ der Schwerpunkte *Flexibilität* und *Kommunikation* Verwendung. Das Pattern „Untersuchung betroffener Subsysteme“ wird neben dem Schwerpunkt *Zuverlässigkeit* ebenfalls in den genannten Gruppen der Schwerpunkte *Flexibilität* und *Kommunikation*

eingesetzt. Beide Patterns liefern für die Ermittlung der Zuverlässigkeitsanforderungen einen geeigneten Überblick über das geplante System und seine vorgegebenen, bereits existierenden Bestandteile. Dadurch liefern sie Basisinformationen, die als Voraussetzung für die Definition konkreter Zuverlässigkeitsanforderungen zu betrachten sind.

Konkrete Zuverlässigkeitsanforderungen sind, wie der Begriff der Zuverlässigkeit selbst, sehr vielfältig. Daher enthält die Gruppe mit acht verschiedenen Patterns eine relativ große Zahl einzelner Muster. Sie befassen sich mit Aspekten der Verfügbarkeit und der problemangemessenen Leistungsfähigkeit ebenso wie mit der Verlässlichkeit der Inhalte verteilter Krankenakten.

Gruppe „Ermittlung der Zuverlässigkeitsanforderungen“:

- Access Control Requirements (Anhang S. 248)
- Approval Requirements (Anhang S. 275)
- Availability Requirements (Anhang S. 292)
- Dynamic Capacity Requirements (Anhang S. 353)
- Response Time Requirements (Anhang S. 516)
- Static Capacity Requirements (Anhang S. 581)
- Throughput Requirements (Anhang S. 589)
- Validation Requirements (Anhang S. 597).

Die Patterns „Availability Requirements“, „Response Time Requirements“, „Dynamic Capacity Requirements“ und „Static Capacity Requirements“ sowie „Throughput Requirements“ helfen bei der Definition von Anforderungen, die die benötigte Verfügbarkeit und Performance des Systems beschreiben. Sie entstammen dem Kapitel „Performance Requirement Patterns“ aus dem Buch „Software Requirement Patterns“ von Stephen Withall (Withall 2007:S. 191–238).

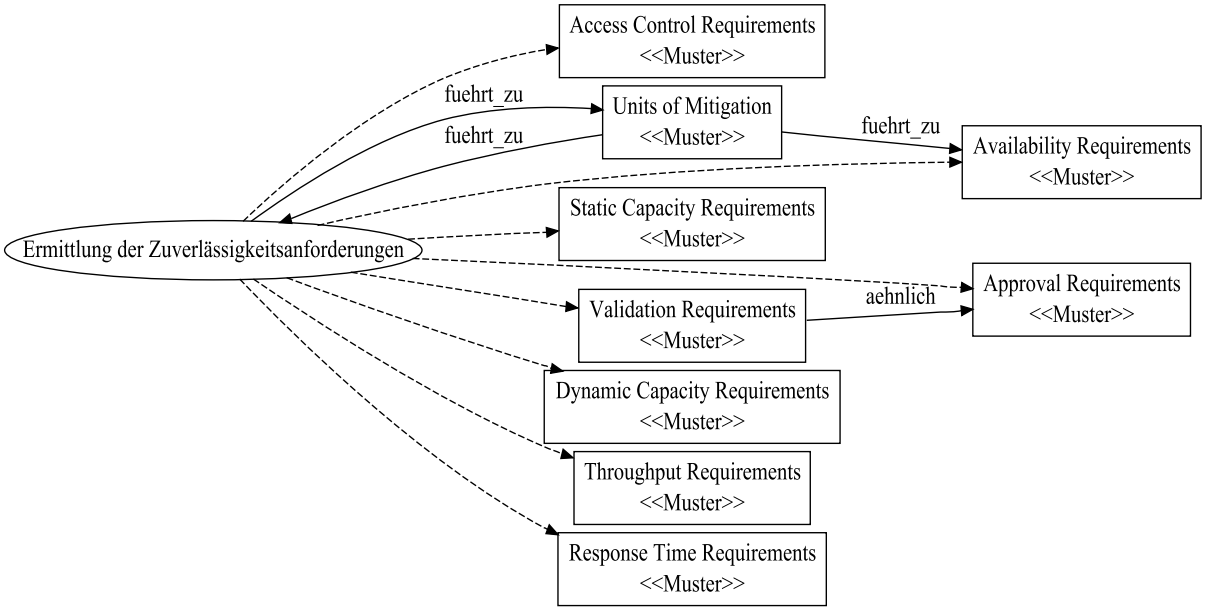



Abbildung 71: Gruppe „Ermittlung der Zuverlässigkeitsanforderungen“ ohne eingehende Beziehungen

Die Abbildung 71 enthält mit dem Pattern „Units of Mitigation“ ein Muster der Ebene *Architektur*. Es dient dazu, die Architektur der verteilten Krankenakte in Einheiten der Fehlerbehandlung zu gliedern. Für jede dieser Einheiten ist eine separate Ermittlung der Zuverlässigkeitsanforderungen erforderlich. Dieser Zusammenhang wird durch die beiden entgegengesetzten Führt-zu-Beziehungen zwischen der Gruppe „Ermittlung der Zuverlässigkeitsanforderungen“ und dem Pattern „Units of Mitigation“ abgebildet.

Zwischen der Gruppe „Ermittlung der Zuverlässigkeitsanforderungen“ und dem Pattern „Units of Mitigation“ liegt eine besondere Abhängigkeit vor (siehe Abbildung 71). Einerseits dienen die Ergebnisse aus der Anwendung der Patterns der Gruppe dazu, die Größe und die Grenzen der Einheiten der Fehlerbehandlung (Units of Mitigation) zu definieren. Andererseits ist für jede definierte Einheit eine separate Ermittlung der Zuverlässigkeitsanforderungen notwendig, um deren interne Gestaltung festzulegen.

Bei bereits im Betrieb befindlichen Systemen kann aus verschiedenen Gründen die Zuverlässigkeit sinken, obwohl das System selbst, unter Berücksichtigung der Maßgaben des Planungszeitpunkts, auf ein bestimmtes Maß an Zuverlässigkeit hin gestaltet ist. Meist sind dafür Veränderungen äußerer Parameter verantwortlich, die über das eingeplante Maß hinausgehen. In einem solchen Fall ist eine Überarbeitung der bestehenden Zuverlässigkeitsanforderungen und ein darauf folgendes Redesign des Systems erforderlich.

Gruppe „Überarbeiten der Zuverlässigkeitsanforderungen“:

- Reassess Overload Decision (Anhang S. 501)
- Revise Procedure (Anhang S. 522).

Die Gruppe „Überarbeiten der Zuverlässigkeitsanforderungen“ enthält dafür zwei verschiedene Patterns, die sich für unterschiedliche Aspekte bei der Überarbeitung von Systemen bezüglich ihrer Zuverlässigkeit eignen. Das Revise-Procedure-Pattern richtet den Fokus auf Zuverlässigkeitsprobleme, die durch die Interaktion zwischen Menschen und Maschinen auftreten und fordert für den Fall von auftretenden Problemen die Überarbeitung von bestehenden Nutzungsrichtlinien.

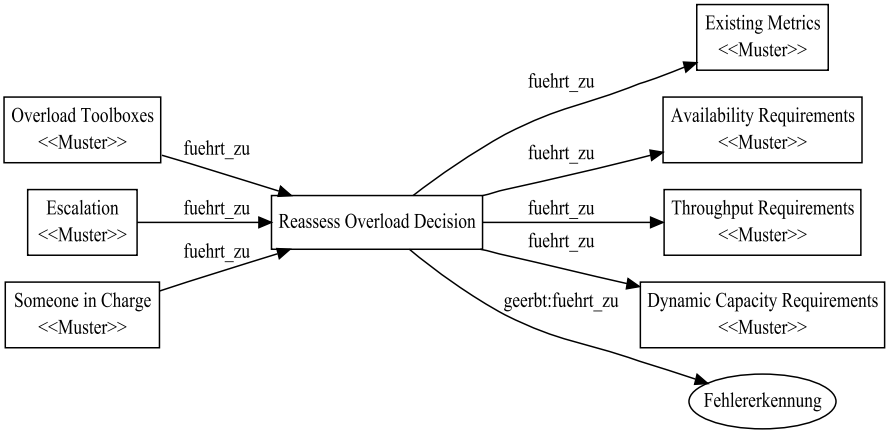



Abbildung 72: „Reassess Overload Decision“ als Beispiel für die Gruppe „Überarbeiten der Zuverlässigkeitsanforderungen“

Das Reassess-Overload-Decision-Pattern ist das Muster, das für die Behebung von Zuverlässigkeitsproblemen bestimmt ist, die durch veränderte Rahmenbedingungen entstehen. Unter

veränderten Rahmenbedingungen ist hierbei ein umfangreiches Gebiet von Veränderungen zu verstehen, das von stark gestiegenen Nutzer- oder Nutzungszahlen über sprunghaft ansteigende Speicherbedarfe bis hin zu plötzlich auftretenden Zusatzbelastungen reicht. Die Anwendung des Reassess-Overload-Decision-Patterns ist dabei, wie in Abbildung 72 dargestellt, das Resultat aus einer schrittweisen Eskalation der Verfahren zur Behebung eines auftretenden Zuverlässigkeitsproblems und mündet am Ende in einer Anpassung der Zuverlässigkeitsanforderungen.

### 5.4.2. Architektur

Die Ebene *Architektur* im Schwerpunkt *Zuverlässigkeit* beschreibt eine Sammlung von Patterns zur Umsetzung der, mittels der Patterns der Ebene *Anforderungsanalyse* definierten Anforderungen auf der Ebene der Software- bzw. der Systemarchitektur. Sie umfasst eine umfangreiche Sammlung von Patterns und ist in vier Gruppen gegliedert. Ziel der Gruppen und enthaltenen Patterns ist es, die Anforderungen bezüglich der Verlässlichkeit, Fehlertoleranz und Verfügbarkeit zu erfüllen. Dazu beschreiben die Patterns, wie durch die Erfüllung der durch die Anforderungen definierten Teilziele die erforderliche Zuverlässigkeit für das gesamte System erreicht werden kann.

Gruppen:

- Architekturpatterns Fehlertoleranz
- Patientenidentifikation
- Suche im verteilten Dokumentensystem
- Verbindung zwischen Anwendungslogik und Daten

Die umfangreichste Gruppe der Ebene *Architektur* ist die Gruppe der „Architekturpatterns Fehlertoleranz“. Sie enthält elf Patterns, die an unterschiedlichen Stellen die Zuverlässigkeit des Gesamtsystems einer verteilten Krankenakte durch erhöhte Toleranz gegenüber verschienen Arten von Fehlern verbessern. Ihre Patterns beschränken sich dabei nicht nur auf die Softwarearchitektur des Systems, sondern berücksichtigen auch Aspekte der Systemarchitektur im Allgemeinen sowie organisatorische Vorkehrungen zur Vermeidung oder Behandlung von auftretenden Fehlern. Inhaltlich orientiert sich die Gruppe stark an dem Kapitel „Architectural Patterns“ aus „Patterns for Fault Tolerant Software“ (Hanmer 2007:S. 33–81), da diese Quelle eine umfassende Darstellung der Architekturaspekte zu Fehlertoleranz beinhaltet.

Gruppe „Architekturpatterns Fehlertoleranz“:

- Correcting Audits (Anhang S. 331)
- Escalation (Anhang S. 367)
- Fault Observer (Anhang S. 380)
- Maintenance Interface (Anhang S. 431)
- Maximize Human Participation (Anhang S. 435)
- Minimize Human Intervention (Anhang S. 449)
- Recovery Blocks (Anhang S. 503)
- Redundancy (Anhang S. 504)

- Software Update (Anhang S. 575)
- Someone in Charge (Anhang S. 576)
- Units of Mitigation (Anhang S. 591).

Eines der zentralen Patterns der Gruppe ist das Pattern „Units of Mitigation“. Es beschreibt die Gliederung des gesamten Systems in Blöcke oder Einheiten der Fehlerbehandlung. Eine „Unit of Mitigation“ beschreibt somit den maximalen Bereich, auf den sich ein innerhalb dieser Einheit aufgetretener Fehler im System auswirken darf. Die aus der Abbildung 73 ersichtliche Beziehung zwischen „Units of Mitigation“ und „Redundancy“ zeigt auf, dass ein solcher Fehler, wenn er zum Ausfall einer ganzen Einheit führt, durch die Redundanz dieser Einheit kompensiert werden kann.

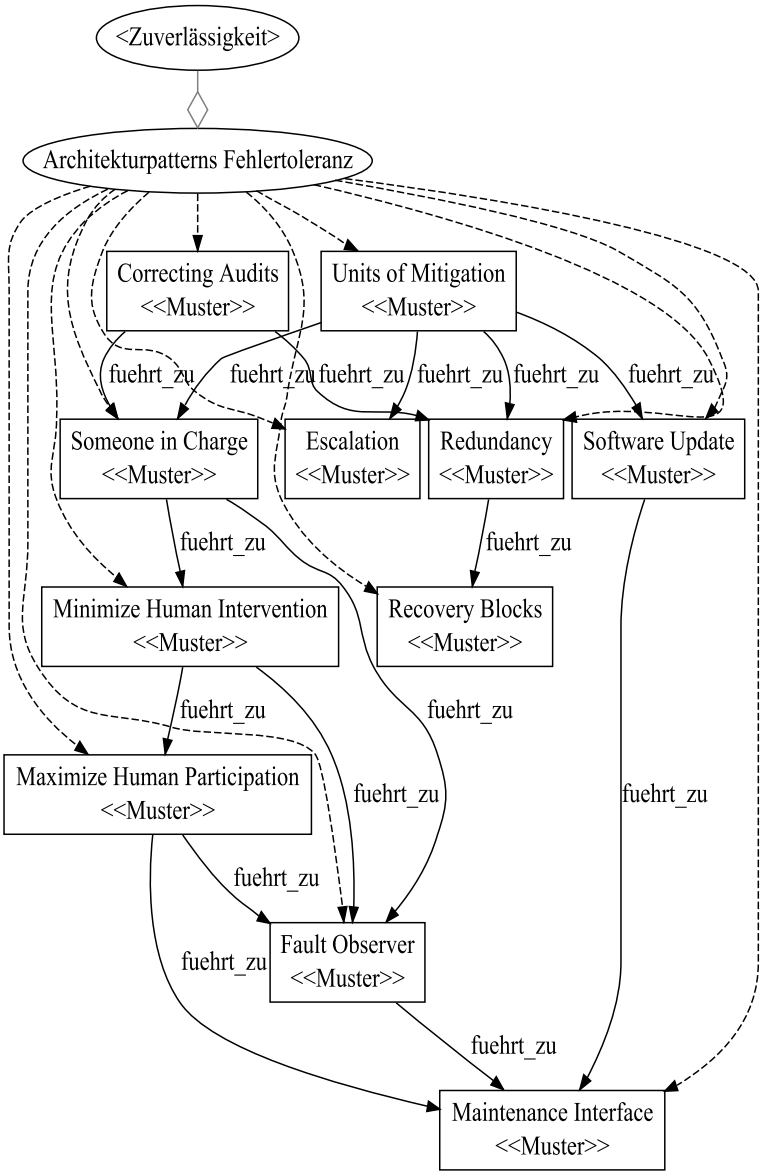



Abbildung 73: Gruppe „Architekturpatterns Fehlertoleranz“ mit internen Abhängigkeiten

Außerdem ist das Pattern „Units of Mitigation“, wie auf Seite 191 erläutert, das zentrale Element im zweistufigen Prozess der Analyse der Zuverlässigkeitsanforderungen. Es führt durch seine Beziehungen, nach einer Phase der ersten Architekturdefinition, zu einer

nochmaligen, feingranulareren Definition der Zuverlässigkeitsanforderungen für jede einzelne „Unit of Mitigation“.

Das Pattern „Recovery Blocks“ besitzt bei unzureichend genauer Betrachtung eine gewisse Ähnlichkeit zum Pattern „Units of Mitigation“. Es beschreibt einen Lösungsansatz zur Kompensation von systematischen, z. B. durch den verwendeten Algorithmus bedingten Fehlern mittels der Verwendung redundanter Funktionen mit verschiedenen gleichwertigen Implementierungen innerhalb einer „Unit of Mitigation“ (vgl. Hanmer 2007:S. 53–54). „Recovery Blocks“ können in diesem Kontext unterschiedlich eingesetzt werden. Einerseits ist es verbreitet sie parallel zur eigentlichen Implementierung auszuführen, um mit ihren Ergebniswerten deren Lösung zu validieren und ggf. auch darin enthaltene Fehler zu korrigieren. Andererseits können „Recovery Blocks“ auch als erste Eskalationsstufe (vgl. Hanmer 2007:S. 55) nach dem Scheitern eines Algorithmus und somit als erster Kompensationsversuch zur Vermeidung eines kritischen Fehlers verwendet werden.

Die schrittweise Auswahl von Fehlerbehandlungs- und Kompensationsmechanismen wird durch das Escalation-Pattern beschrieben. Die in ihm zur Fehlerbehandlung vorgegebenen Stufen oder Schritte sind von einfachen und wenig aufwendigen Methoden hin zu immer drastischeren Methoden der Fehlerbehandlung sortiert (vgl. Hanmer 2007:S. 71). Das im vorherigen Absatz vorgestellte Pattern „Recovery Blocks“ beschreibt nur eine mögliche Eskalationsstufe. Weitere mögliche Methoden sind der Rollback oder Reset von Daten (vgl. S. 533 - Rollback und S. 337 - Data Reset) oder im Fall nicht automatisch behandelbarer Fehler die Intervention durch eine technisch dafür ausgebildete Person.

Die Umsetzung der Eskalation mit ihrem Weg bis zu einer Beteiligung von Personen an der Fehlerkompensation wird durch die Patternsequenz „Someone in Charge“, „Minimize Human Intervention“ und „Maximize Human Participation“ beschrieben. Abbildung 73 zeigt die sequenzielle Beziehungssituation der drei Patterns. Das Someone-in-Charge-Pattern erklärt, dass zur Umsetzung des Escalation-Patterns für jede „Unit of Mitigation“ eine eindeutig definierte Komponente zur Behandlung von Fehlern existieren muss (vgl. Hanmer 2007:S. 67). Sie kann von mehreren „Units of Mitigation“ gemeinsam genutzt werden oder auch für jede „Unit of Mitigation“ einzeln existieren. Sie ist die Basis für sowohl die Umsetzung des Escalation-Patterns als auch verschiedener anderer Mechanismen der Fehlerbehandlung.

Das Pattern „Minimize Human Intervention“ beschreibt ein Grundprinzip für die Fehlerbehandlung, das auch direkten Einfluss auf die Gestaltung von Eskalationsstufen besitzt. Es besagt, dass es vorteilhaft ist, ein System so zu konzeptionieren, dass Fehler so lange als möglich von automatischen Mechanismen kompensiert werden. Diese Forderung gründet auf der Tatsache, dass Menschen dazu tendieren Fehler zu machen (vgl. Hanmer 2007:S. 57). Erwartete und bekannte Fehler können durch automatisierte Methoden behandelt werden. Treten Fehler auf, deren Ursachen unbekannt sind und für die keine Kompensationsmethoden verfügbar sind, so können diese nicht ohne das Zutun von Experten behoben werden. Diesen Zusammenhang beschreibt das Pattern „Maximize Human Participation“. Es setzt die Verfügbarkeit geeigneter Schnittstellen zur Information über vorliegende Fehler und zur Behebung dieser Fehlern durch Menschen voraus. Potenzielle Kandidaten für die Umsetzung

solcher Schnittstellen sind die Pattern „Maintenance Interface“ oder „Fault Observer“ (vgl. Hanmer 2007:S. 61).

Das Pattern „Fault Observer“ ist eine spezielle Ausprägung des Observer-Patterns (vgl. S. 461). Es dient als Datensenke für Informationen über auftretende Fehler und so der Überwachung des Systemzustands. Sein Einsatzzweck reicht von der einfachen Entkopplung zwischen der Anwendungslogik und der Fehlerbehandlung bis hin zur Umsetzung eines zentralisierten Fehlerlogs oder einer zentralisierten Fehlerbehandlung.

Bezogen auf die Konzeption verteilter Krankenakten lassen sich durch die Anwendung der Patterns der Gruppe „Architekturpatterns Fehlertoleranz“ sowohl die Nachvollziehbarkeit aufgetretener Fehler als auch die Vermeidung von Fehlinformation oder Systemausfällen durch das Auftreten allgemeiner Fehlern erlangen.

Neben der Fehlertoleranz gibt es bei verteilten Krankenakten weitere die Zuverlässigkeit beeinflussende Faktoren. Einer dieser Faktoren ist die Zuverlässigkeit der Identifikation von Patienten innerhalb des Systems, denn die Zuverlässigkeit der verteilten Krankenakte umfasst auch die Verlässlichkeit der in ihr zur Verfügung gestellten Informationen. Das bedeutet unter anderem, dass nur Informationen zum richtigen Patienten in mindestens aufgabenbezogener Vollständigkeit zur Verfügung stehen müssen. Eine Verwechslung von Personen oder ein versehentliches Zuordnen einzelner Ergebnisse zu falschen Personen würde die Verlässlichkeit der Inhalte zerstören. Da eine verteilte Krankenakte auf einer Vielzahl verteilt existierender Datenquellen basiert, sind zuverlässige Mechanismen zur Patientenidentifikation wichtig für die Zuverlässigkeit der verteilten Krankenakte.

Gruppe „Patientenidentifikation“:

- Master Patient Index (Anhang S. 433)
- Unified Healthcare Identifier (Choi u. a. 2006:S. 44; Bergmann 2006:S. 51).

In der Literatur zu elektronischen Krankenakten werden für die einrichtungsübergreifende Kommunikation von Patientendaten und das daraus resultierende Problem der Identifikation von Patienten über die Grenzen von Softwaresystemen hinaus zwei grundsätzliche Ansätze beschrieben (vgl. z. B. Bergmann 2006:S. 51). Diese Ansätze sind in der Gruppe „Patienten-identifikation“ als allgemein verwendbare Patterns beschrieben. Das Pattern „Unified Healthcare Identifier“ (UHID) beschreibt dabei den Ansatz, jedem Patienten oder sogar jeder an den Prozessen der Patientenversorgung beteiligten Entität, einschließlich des Patienten, eine global eindeutige Identifikationsnummer zuzuordnen. Auf diese Weise werden Patienten durch ihre eindeutige Nummer über die Grenzen der Systeme und Einrichtungen hinweg ein-deutig identifizierbar. Das Pattern „Master Patient Index“ beschreibt einen anderen Ansatz. Es basiert auf einem zentralen Index, der die jeweils lokalen Patienten-IDs einer globalen ID zuordnet. Auf diese Weise kann mittels des zentralen Index zwischen den jeweils lokalen Patienten-IDs übersetzt werden.

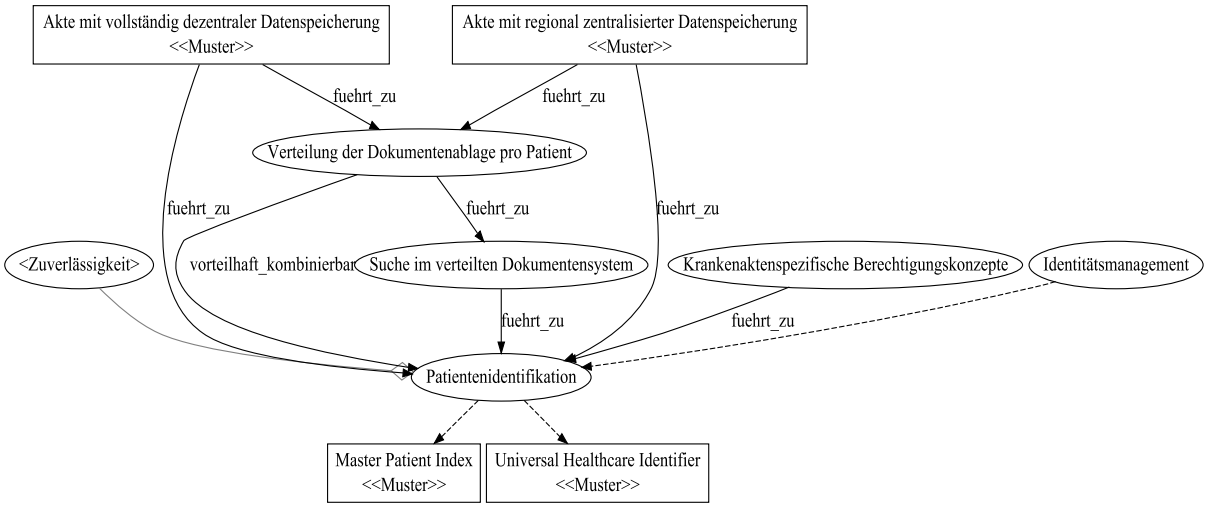



Abbildung 74: Direkte Beziehungen der Gruppe „Patientenidentifikation“

Abbildung 74 zeigt die eingehenden Abhängigkeiten der Gruppe „Patientenidentifikation“. Die dargestellten Beziehungen der Patterns „Akte mit vollständig dezentraler Datenspeicherung“ und „Akte mit regional zentralisierter Datenspeicherung“ sowie die Beziehung zur Gruppe „Verteilung der Dokumentenablage pro Patient“ repräsentieren die besondere Bedeutung der eindeutigen Identifizierbarkeit eines Patienten für Aktenarchitekturen mit verteilter Speicherung der Inhalte. Bei Akten mit vollständig zentraler Speicherung ist die Lösung des Problems der eindeutigen Zuordnung zwischen Patienten und ihren Akteninhalten durch die nicht redundante Verwaltung der Patientenstammdaten einfacher. Deshalb ist in diesem Fall die Anwendung eines Patterns der Gruppe „Patientenidentifikation“ nicht notwendig.

Auch für die Suche in einer verteilten Krankenakte ist die eindeutige Identifizierbarkeit von Patienten wichtig. Hier ist sie unter anderem notwendige Voraussetzung für das zuverlässige Zusammenführen der verteilt gespeicherten Aktenbestandteile eines Patienten. In diesem Kontext beeinflusst auch der für die Suche in einer verteilten Krankenakte ausgewählte Mechanismus die Zuverlässigkeit der Akte. Von ihm hängen sowohl die Vollständigkeit als auch die Qualität der Selektion der Suchergebnisse ab.

Gruppe „Suche im verteilten Dokumentensystem“:

- Flooding based Search (Anhang S. 387)
- Zentraler Suchdienst (Anhang S. 612).

Neben der Zuverlässigkeit beeinflussen die Patterns der Gruppe „Suche im verteilten Dokumentensystem“ auch die Flexibilitätseigenschaften der verteilten Krankenakte. Aus diesem Grund ist die Gruppe bereits auf S. 143 hinsichtlich ihrer Auswirkungen auf die Flexibilität des Aktensystems beschrieben. Abbildung 75 zeigt, dass die Gestaltung der Suche im verteilten Dokumentensystem von der „Verteilung der Dokumentenablage pro Patient“ abhängig ist. Außerdem verdeutlicht die Abbildung, dass die Suche in einem verteilten Dokumentensystem einen geeigneten Mechanismus zur Patientenidentifikation erfordert und sowohl dem Schwerpunkt *Flexibilität* als auch dem Schwerpunkt *Zuverlässigkeit* zugeordnet ist.

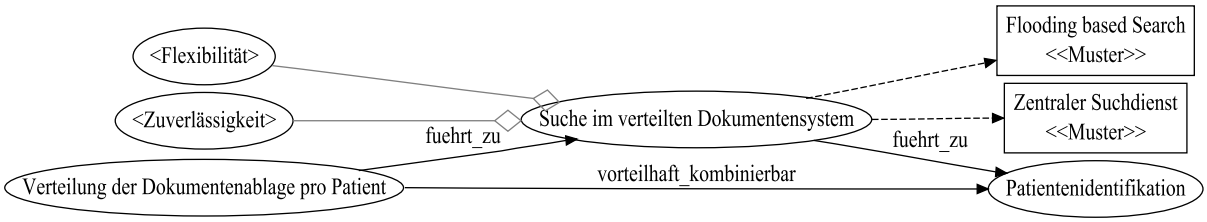



Abbildung 75: Gruppe „Suche im verteilten Dokumentensystem“

Was die Auswirkungen der Gestaltung des Suchmechanismus auf die Zuverlässigkeit einer verteilten Krankenakte anbelangt, lassen sich die folgenden Aspekte festhalten. Während bei ausschließlich Flooding-basierten Suchmechanismen nicht sichergestellt ist, dass die Ergebnismenge einer Suche tatsächlich vollständig ist, kann ein zentraler Suchdienst, wenn er nicht redundant vorgehalten wird, als Single-Point-of-Failure durch seinen Ausfall zu einem Ausfall der gesamten Suchfunktionalität der verteilten Akte führen. Flooding ist aufgrund seiner teilweise für den Benutzer transparent auftretenden Fehler nur für spezielle Einsatzbereiche innerhalb einer Peer-to-Peer-Architektur verwendbar oder muss um zusätzliche, die Zuverlässigkeit verbessernde Maßnahmen ergänzt werden. Beim Einsatz eines zentralen Suchdienstes sind mit höherer Zuverlässigkeit qualitativ hochwertige Suchergebnisse ermittelbar. Um das Problem des Single-Point-of-Failure zu mindern, sollte durch eine redundante Auslegung des Dienstes, dessen Verfügbarkeit erhöht und somit die Zuverlässigkeit der verteilten Akte insgesamt gesteigert werden.

Ein weiterer Aspekt, der die Zuverlässigkeit von Softwaresystemen im Allgemeinen beeinflusst, ist die Art, in der die Kopplung zwischen der Anwendungslogik und dem tatsächlichen Ort der Datenspeicherung erfolgt. Bei der Speicherung von Daten kann es zu einer Verfälschung von Daten durch nicht sachgemäß durchgeführte Änderungen kommen. Solche verfälschten oder möglicherweise verfälschten Daten führen dazu, dass das gesamte System als nicht zuverlässig eingestuft werden muss. Mit dem Aspekt der Speicherung und Änderung von Daten am Ort ihrer Speicherung befassen sich die Patterns der Gruppe „Verbindung zwischen Anwendungslogik und Daten“.

Gruppe „Verbindung zwischen Anwendungslogik und Daten“:

- Database Access Layer (Anhang S. 338)
- Data Access Object (DAO) (Anhang S. 333).

Das Architekturpattern „Database Access Layer“ dient vorrangig der Entkopplung von verschiedenen Aspekten innerhalb der Softwarearchitektur der verteilten Krankenakte und damit der Erhöhung der Flexibilität der Softwarearchitektur. Ein erwünschter Nebeneffekt einer eindeutig definierten Datenzugriffsschicht ist, dass alle weiteren Komponenten eines Softwaresystems den Zugriff auf Daten nicht durch direkten Zugriff auf den Speicherort, sondern durch die Verwendung von Zugriffskomponenten der „Database Access Layer“ durchführen. Die Komponenten dieser Schicht sind somit der einzige Zugriffsweg auf die Daten. Sie können folglich an zentraler Stelle Konsistenzbedingungen und eine transaktionale Verarbeitung etc. durchsetzen und so die Zuverlässigkeit bei der Datenspeicherung verbessern.

### 5.4.3. Design

Die Patterns der Ebene *Design* dienen der Gestaltung der Details in einer bereits durch die Anwendung von Architekturpatterns vordefinierten Anwendungs- und Softwarearchitektur. Innerhalb des Schwerpunkts *Zuverlässigkeit* beschreiben sie verschiedene Lösungsansätze zur konkreten Umsetzung von Mechanismen zur Erkennung, Behebung, Kompensation und Vermeidung von Fehlern und somit zur Erreichung einer insgesamt erhöhten Fehlertoleranz des gesamten Softwaresystems einer verteilten Krankenakte. Das Anwendungsgebiet der Patterns liegt dabei in der konkreten Umsetzung der Zuverlässigkeitsanforderungen (vgl. S. 190) im Rahmen der bereits definierten Architekturbestandteile der verteilten Krankenakte. Im Schwerpunkt *Zuverlässigkeit* enthält die Ebene *Design* die folgenden acht Gruppen:

- Eingabevalidierung
- Fehlererkennung
- Fehlerbehandlung
- Fehleranalyse
- Fehlerkorrektur
- Automatisierte Fehlerbehebung
- Recovery Types
- Fehlerkompensation.

Die Eingabevalidierung stellt den Versuch dar, bereits zum Zeitpunkt der Datenerfassung oder generell bei der Interaktion von Menschen mit Softwaresystemen Fehler durch aktive Prüfung der Gültigkeit von eingegebenen Werten zu vermeiden. Dazu wird die Prüfung eingegebener Werte häufig von einer direkten noch während der Eingabe durchgeführten Benachrichtigung des Benutzers über die Ungültigkeit seiner getätigten Eingabe, einschließlich gegebenenfalls nötiger Korrekturhinweise, begleitet.

Gruppe „Eingabevalidierung“:

- Approval Workflow (Anhang S. 277)
- Data and Rule based Validation (Anhang S. 338)
- Regular Expression based Validation (Anhang S. 507).

Von den, in der Gruppe Eingabevalidierung enthaltenen drei Patterns basieren die beiden Patterns „Data and Rule based Validation“ und „Regular Expression based Validation“ ausschließlich auf technischen Vorkehrungen. Sie orientieren sich konzeptionell an den Ideen aus vergleichbaren Validierungskomponenten der Frameworks ASP.NET und JSF. Bei der „Regular Expression based Validation“ wird ausschließlich eine Regel (regulärer Ausdruck) auf den eingegebenen Wert angewendet, um dessen Gültigkeit zu prüfen. Dieses Verfahren wird zur Prüfung der Gültigkeit des Formats der eingegebenen Zeichenkette genutzt und findet in der Praxis z. B. bei der Prüfung von eingegebenen E-Mail-Adressen, Postleitzahlen oder Telefonnummern Verwendung. Bei der „Data and Rule based Validation“ werden zur Gültigkeitsprüfung eines eingegebenen Wertes neben dem Wert selbst und einer Regel zusätzlich weitere Daten z. B. für einen Abgleich mit gültigen Werten oder zur Berechnung von Vergleichswerten herangezogen. Hierbei handelt es sich sowohl um den aufwendigeren

als auch den flexibleren Ansatz zur Validierung von Eingaben. Er eignet sich bei verteilten Krankenakten z. B. zur Validierung von eingegebenen ICD- oder LOINC-Codes (vgl. S. 75 ff.) oder zur Prüfung auf der Basis von Abhängigkeiten zwischen mehreren eingegebenen Messwerten.

Einen anderen Weg beschreitet das Pattern „Approval Workflow“. Es beschreibt die Möglichkeit einen eingegebenen Wert oder eine Sammlung eingegebener Werte nach ihrer Eingabe zum Zweck der Validierung an eine zweite, sachkundige Person weiterzuleiten und vor der weiteren Verarbeitung überprüfen zu lassen. Zu diesem Zweck wird nach der Eingabe ein Freigabeworkflow angestoßen, der sicherstellt, dass die Daten erst nach ihrer Überprüfung und der darauf folgenden Freigabe vom System verwendet werden. Ein typischer Anwendungsfall für dieses Pattern ist, wenn auf Befunden oder Arztbriefen zwei oder mehrere Ärzte durch ihre Unterschrift die Korrektheit des Inhalts bestätigen.

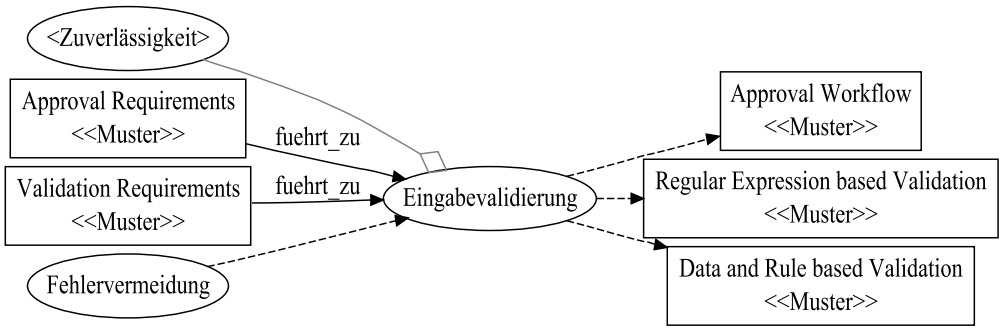



Abbildung 76: Gruppe „Eingabevalidierung“

Die Abbildung 76 zeigt, dass die Auswahl und Gestaltung der Mechanismen der Eingabevalidierung im Wesentlichen von den Ergebnissen der Patterns „Approval Requirements“ und „Validation Requirements“ abhängig sind. Die Eingabevalidierung ist eine Art der Früherkennung von potenziellen Fehlern, noch bevor sie tatsächlich einen Fehler in einem Softwaresystem auslösen oder einen inhaltlichen Fehler in gespeicherten Daten verursachen können. Auf diese Weise lassen sich nicht alle möglichen Fehler verhindern. Zum einen sind nicht alle möglichen Arten von Fehlern auf Mensch-Maschine-Interaktion zurückzuführen; zum anderen berücksichtigen die Regeln der Eingabevalidierung meist nicht alle möglichen Fehler. Aus diesem Grund sind in der Gruppe „Fehlererkennung“ 16 Patterns zur Erkennung von Fehlern, auch außerhalb des Kontexts der Mensch-Maschine-Interaktion zusammengefasst.

Gruppe „Fehlererkennung“:

- Acknowledgement (Anhang S. 251)
- Checksum (Anhang S. 314)
- Complete Parameter Checking (Anhang S. 322)
- Error Containment Barrier (Anhang S. 363)
- Existing Metrics (Anhang S. 368)
- Fault Correlation (Anhang S. 378)
- Heartbeat (Anhang S. 399)

- Leaky Bucket Counter (Anhang S. 421)
- Realisitc Threshold (Anhang S. 499)
- Riding over Transients (Anhang S. 523)
- Routine Audits (Anhang S. 535)
- Routine Exercises (Anhang S. 537)
- Routine Maintenance (Anhang S. 538)
- System Monitor (Anhang S. 584)
- Voting (Anhang S. 603)
- Watchdog (Anhang S. 605).

Die Fehlererkennung ist, wie in Abbildung 77 dargestellt, notwendige Voraussetzung für eine automatisierte Fehlerbehebung, Fehlerbehandlung und Fehlerkompensation und deshalb ein zentraler Bestandteil des Aspekts *Fehlertoleranz*. Die in der Abbildung innerhalb der Gruppe dargestellten Beziehungen entsprechen den in (Hanmer 2007:S. 87) publizierten Beziehungen.

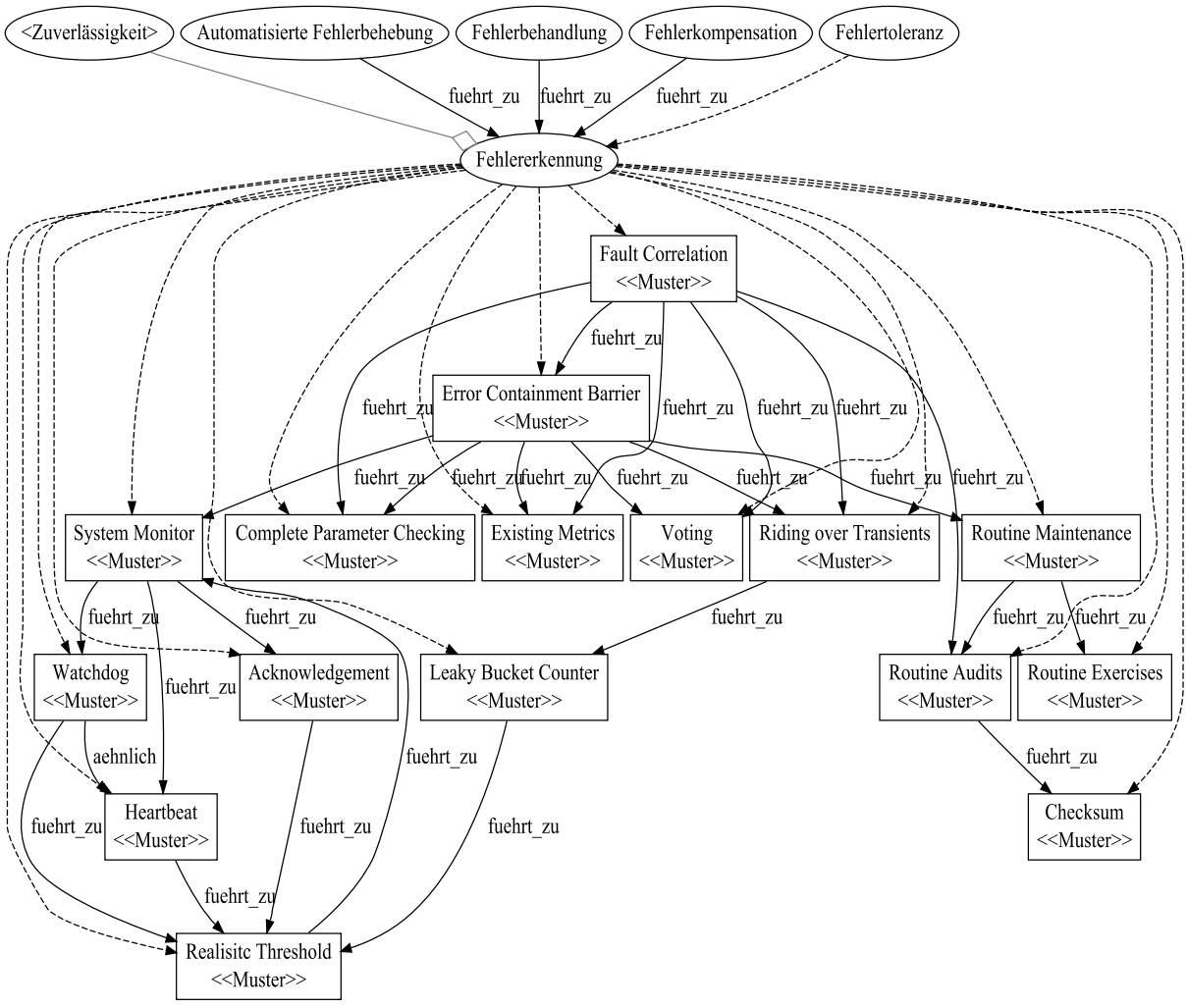



Abbildung 77: Gruppe „Fehlererkennung“ mit internen Abhängigkeiten

Das Spektrum der in der Gruppe „Fehlererkennung“ enthaltenen Patterns reicht von der Erkennung von Fehlern aus der technischen Infrastruktur bis hin zur Erkennung von Fehlern in den organisatorischen Abläufen. Fehler aus der technischen Infrastruktur lassen sich z. B.

mit den Patterns „Heartbeat“, „Checksum“ und „Acknowledgement“ erkennen. Organisatorische Fehler, die z. B. aus fehlerhaften Abläufen bei der Fehlerbehandlung oder dem fehlerhaften Umgang mit dem System resultieren, lassen sich z. B. durch „Routine Exercises“ und „Routine Audits“ ermitteln. Abhängig von der Art des zu erkennenden Fehlers bietet die Gruppe verschiedene Patterns zur Fehlererkennung. Eine ausführliche Beschreibung der Patterns dieser Gruppe und ihrer Beziehungen befindet sich im Kapitel „Detection Patterns“ im Buch „Patterns for Fault tolerant Systems“ (Hanmer 2007:S. 83–137).

Die Erkennung von Fehlern ist ein zentraler Schritt zur Erreichung von Fehlertoleranz. Sie allein ist aber nicht ausreichend, um ein System zu schaffen, das auch beim Auftreten von Fehlern weiter zuverlässig verwendbar bleibt. Dazu sind neben der Fehlererkennung auch Maßnahmen zur Behandlung von Fehlern als Reaktion auf die erkannten Fehler sowie Maßnahmen zur dauerhaften Beseitigung der Fehler erforderlich.

Gruppe „Fehlerbehandlung“:

- Gruppe „Fehleranalyse“
- Gruppe „Fehlerkorrektur“
- Gruppe „automatisierte Fehlerbehebung“
- Gruppe „Fehlerkompensation“.

Um einen Fehler korrigieren, also dauerhaft beseitigen zu können, ist es notwendig, seine Ursache zu kennen. Die dabei typische Vorgehensweise ist laut Hanmer (Hanmer 2007:S. 227) die Fehlerverifikation, gefolgt von einer Fehlerdiagnose, aus deren Ergebnissen dann eine Lösung zur Fehlerkorrektur abgeleitet wird. Die Patterns zu den Schritten Fehlerverifikation und Fehlerdiagnose sind in der Gruppe „Fehleranalyse“ zusammengefasst. Sie beschreiben Ansätze um aus einem erkannten, systematisch auftretenden Fehler die notwendigen Informationen für dessen dauerhafte Korrektur abzuleiten.

Gruppe „Fehleranalyse“:

- Reproducible Error (Anhang S. 515)
- Root Cause Analysis (Anhang S. 534)
- Revise Procedure (Anhang S. 522).

Eine dauerhafte Korrektur kann nur für reproduzierbare Fehler gewährleistet werden. Deshalb ist es notwendig, vor der detaillierten Analyse eines Fehlers zu überprüfen, ob es sich um einen reproduzierbaren Fehler handelt. Dazu kann nach dem Pattern „Reproducible Error“ vorgegangen werden. Liegt ein reproduzierbarer Fehler vor, ist im nächsten Schritt der Fehleranalyse die eigentliche Fehlerursache festzustellen, denn ein identifizierter Fehler kann immer auch Folgefehler eines noch nicht erkannten anderen Fehlers sein. Das Pattern „Root Cause Analysis“ bietet hierfür Empfehlungen zur Ermittlung der tatsächlichen Fehlerursache.

Normalerweise sollen technische Mitarbeiter, die im Fehlerfall zur Behandlung des Fehlers hinzugezogen werden, die Dauer, in der ein System nicht verfügbar ist, reduzieren. Ist das nicht der Fall, oder verlängert das Eingreifen dieser Mitarbeiter die Phase sogar noch, so

sollte mittels des Patterns „Revise Procedure“ eine Analyse und Überarbeitung der technischen Dokumentation und insbesondere der Verfahrensanweisungen für die Fehlerbehebung durchgeführt werden.

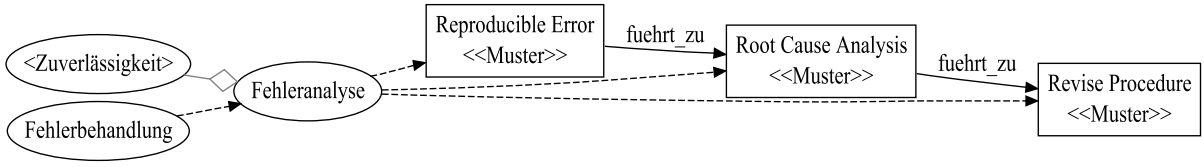



Abbildung 78: Gruppe „Fehleranalyse“ mit internen Abhängigkeiten

Um einen Fehler dauerhaft zu beheben, muss auf die Analyse des Fehlers eine Korrektur des Fehlers folgen, sofern die Analyse einen korrigierbaren Fehler ergeben hat. Die Gruppe „Fehlerkorrektur“ enthält drei Patterns zur Bewertung der Korrekturrelevanz und zur dauerhaften Behebung softwarebedingter Fehler. Unter softwarebedingten Fehlern sind hierbei sowohl Fehler, die sich im Code des Softwareprodukts befinden, als auch Unzulänglichkeiten im Code, die zu Fehlern führen können, zu verstehen.

Gruppe „Fehlerkorrektur“:

- Let Sleeping Dogs Lie (Anhang S. 423)
- Reintegration (Anhang S. 507)
- Small Patches (Anhang S. 574).

Das Pattern „Let Sleeping Dogs Lie“ dient der Bewertung des Nutzens einer Korrektur des Programmcodes im Verhältnis zu den möglichen Risiken. Die beiden anderen Patterns, „Reintegration“ und „Small Patches“ befassen sich mit der Art und Weise der Einbringung von Fehlerkorrekturen in bestehende Systeme. Das Pattern „Small Patches“ empfiehlt hierzu, Änderungen immer möglichst klein zu halten, um die Wahrscheinlichkeit von unerwarteten negativen Auswirkungen auf andere Bestandteile des Systems zu minimieren. Das Reintegration-Pattern beschreibt allgemein die Notwendigkeit definierter Prozesse zur Reintegration von korrigierten Softwarekomponenten in laufende Systeme.

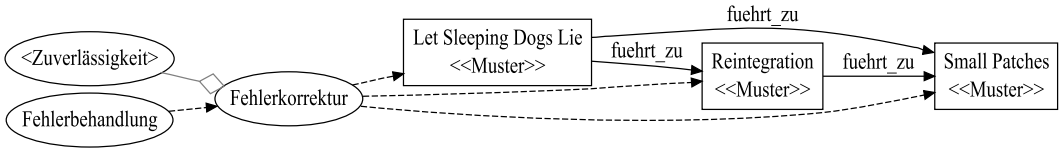



Abbildung 79: Gruppe „Fehlerkorrektur“ mit internen Abhängigkeiten

Die Abbildung 79 zeigt die Fehlerkorrektur als Teilbereich der Fehlerbehandlung und stellt die Abhängigkeiten zwischen den enthaltenen Patterns dar. Das Pattern „Let Sleeping Dogs Lie“ hat dabei zwei ausgehende Führt-zu-Beziehungen. Diese Beziehungen bilden ab, dass die Anwendung der beiden anderen Patterns der Gruppe immer dann sinnvoll ist, wenn die Korrekturwürdigkeit eines Fehlers festgestellt ist. Die Beziehung zwischen den Patterns „Reintegration“ und „Small Patches“ bringt zum Ausdruck, dass ein definierter Prozess für die Reintegration von Softwarekomponenten von einer Beschränkung auf möglichst kleine Änderungen profitieren kann.

Der Prozess der Fehlererkennung, Fehleranalyse und Fehlerkorrektur ist komplex, erfordert die Mitwirkung von Entwicklern und nimmt Zeit in Anspruch. Deshalb ist dieser Weg nicht für die Anwendung im Rahmen von Sofortmaßnahmen zur Erhaltung der Betriebsfähigkeit eines Systems geeignet. Dazu sind Maßnahmen notwendig, die ohne exakte Kenntnis der Fehlerursache durch definiertes Verhalten die Lauffähigkeit des Systems oder die Konsistenz der Daten erhalten können. Die Gruppe „Automatisierte Fehlerbehebung“ enthält die folgenden 14 Patterns.

Gruppe „Automatisierte Fehlerbehebung“:

- Checkpoint (Error recovery) (Anhang S. 312)
- Concentrated Recovery (Anhang S. 329)
- Data Reset (Anhang S. 337)
- Error Handler (Anhang S. 366)
- Failover (Anhang S. 377)
- Individuals Decide Timing (Anhang S. 405)
- Limit Retries (Anhang S. 424)
- Quarantine (Anhang S. 496)
- Remote Storage (Anhang S. 514)
- Restart (Anhang S. 518)
- Return to Reference Point (Anhang S. 519)
- Roll-Forward (Anhang S. 532)
- Rollback (Anhang S. 533)
- What to save (Anhang S. 607).

Eine häufig verwendete Lösung zur Erhaltung der Konsistenz von Daten nach dem Auftreten eines Fehler ist die Anwendung des Patterns „Rollback“. Es beschreibt das Zurücksetzen von Daten auf einen als konsistent bekannten Zustand eines Zeitpunkts vor dem Auftreten des Fehlers. Zur Umsetzung des Patterns „Rollback“ ist die Verwendung verschiedener anderer Patterns notwendig oder zumindest sinnvoll. Ein Rollback-Mechanismus benötigt gesichert valide frühere Systemzustände, auf die das System im Fehlerfall zurückgesetzt werden kann. Dazu kann das Pattern „Checkpoint (Error recovery)“ verwendet werden. Es ermöglicht die Sicherung der letzten validen Systemzustände. Für die Umsetzung eines Rollback-Mechanismus in einer durch redundante Komponenten abgesicherten Umgebung ist zusätzlich zum Pattern „Checkpoint“ auch die Verwendung der Patterns „Remote Storage“ und „What to save“ empfehlenswert. Ihre Anwendung erlaubt beim Ausfall einer der redundant vorgehaltenen Komponenten die automatische Übernahme im Sinne eines Failover mit gleichzeitigem Rollback der möglicherweise nicht vollständig ausgeführten Operation, in der der Fehler aufgetreten ist. Ist die Aufgabe der ausgefallenen Komponente durch ihren Stellvertreter übernommen, kann ihre Funktionsfähigkeit durch beispielsweise einen Neustart (Restart-Pattern) wiederhergestellt werden.

Das Pattern „Quarantine“ beschreibt das Aussondern oder Isolieren von Inhalten oder Softwarebestandteilen, die möglicherweise für das Auftreten eines Fehlers verantwortlich sind. Sie werden in einem Quarantänebereich für eine Fehleranalyse aufbewahrt und nur korrigiert in den normalen Verarbeitungsprozess zurückgeführt. Nach der Aussonderung von

fehlerhaften Daten in einen Quarantänebereich ist es notwendig, im Rahmen der Verarbeitung dieser Daten bereits durchgeführte Änderungen rückgängig zu machen oder deren Auswirkungen zu kompensieren. Diesen Zusammenhang beschreiben die in der Abbildung 80 dargestellten ausgehenden Beziehungen des Patterns „Quarantine“.

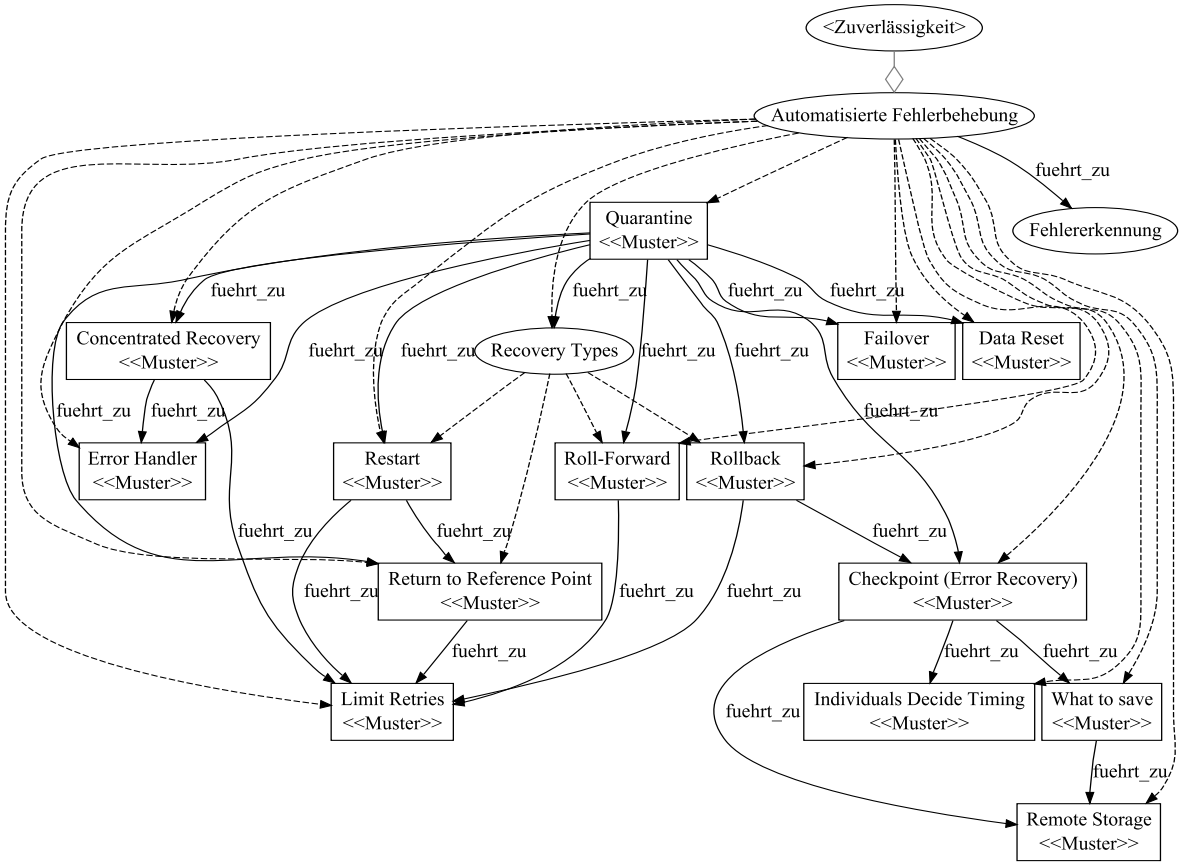



Abbildung 80: Gruppe „Automatisierte Fehlerbehebung“ mit internen Abhängigkeiten

Eine Teilmenge von vier Patterns aus der Gruppe „Automatisierte Fehlerbehebung“ beschreibt Typen von grundsätzlich unterschiedlichen Herangehensweisen zur automatischen Wiederherstellung der Funktionsfähigkeit eines Softwaresystems. Diese vier Patterns sind deshalb zusätzlich in der Gruppe „Recovery Types“, einer echten Teilmenge der automatisierten Fehlerbehebung zusammengefasst. Die Zuordnung der vier Patterns basiert auf der Darstellung von Hanmer in der Language-Map „A Pattern Language for Fault Tolerant Software“ (Hanmer 2007).

Gruppe „Recovery Types“:

- Restart (Anhang S. 518)
- Return to Reference Point (Anhang S. 519)
- Roll-Forward (Anhang S. 532)
- Rollback (Anhang S. 533).

Die Abbildung 81 beinhaltet die Gruppe „Recovery Types“ als Teilmenge der Gruppe „Automatisierte Fehlerbehebung“ und verdeutlicht die Abhängigkeit zwischen dem Pattern „Quarantine“ und den verschiedenen „Recovery Types“. Wird ein Element, das einen Fehler ausgelöst hat, in den Quarantänebereich verschoben, so ist zur Wiederherstellung der Betriebsbereitschaft des Systems einer der „Recovery Types“ anzuwenden. Dieses Vorgehen ist nur notwendig, wenn der Fehler die Betriebsbereitschaft oder die Qualität der Daten des Systems negativ beeinflusst hat.

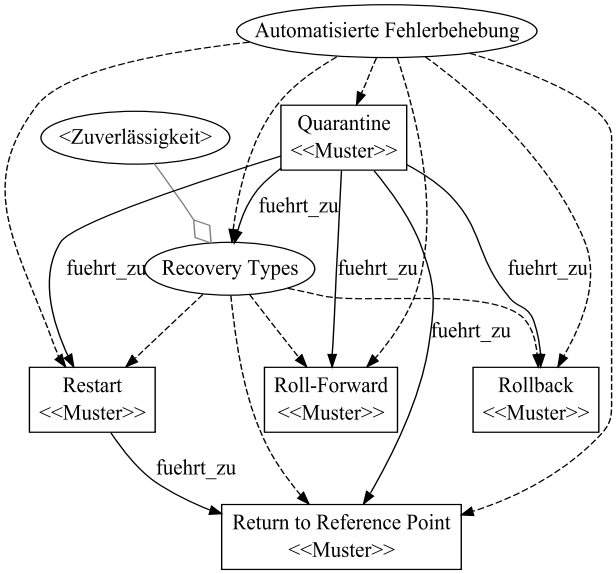



Abbildung 81: Gruppe „Recovery Types“ mit internen Abhängigkeiten

Die bisher beschriebenen Gruppen der Fehlerbehandlung befassen sich mit der Behebung von Fehlern z. B. durch das Zurücksetzen von Datenspeichern auf bekannt gültige Zustände oder das dauerhafte Beheben von Fehlern im Quellcode des Softwaresystems. In einem Softwaresystem, und somit in einer verteilten Krankenakte, können zur Laufzeit weitere Arten von Fehlern auftreten, die nicht sofort behoben werden können. Um das Auftreten dieser Fehler, daraus resultierender Systemausfälle oder sonstiger Engpässe im Systembetrieb verkraften zu können, ist es notwendig, dass eine verteilte Krankenakte zusätzlich Mechanismen zur Kompensation solcher Fehler vorsieht.

Die Gruppe Fehlerkompensation beinhaltet 16 Patterns zur Milderung oder Kompensation der Auswirkungen aufgetretener Fehler. Inhaltlich orientiert sich die Gruppe am Kapitel „Error Mitigation Patterns“ aus „Patterns for Fault Tolerant Systems“ (Hanmer 2007:S. 179 ff.).

Gruppe „Fehlerkompensation“:

- Deferrable Work (Anhang S. 343)
- Equitable Resource Allocation (Anhang S. 362)
- Error Correcting Codes (Anhang S. 365)
- Expansive Automatic Controls (Anhang S. 370)
- Final Handling (Anhang S. 385)
- Finish Work in Progress (Anhang S. 386)
- Fresh Work Before Stale (Anhang S. 390)

- Marked Data (Anhang S. 432)
- Overload Toolboxes (Anhang S. 462)
- Protective Automatic Controls (Anhang S. 483)
- Queue for Ressources (Anhang S. 498)
- Reassess Overload Decision (Anhang S. 501)
- Share the Load (Anhang S. 558)
- Shed Load (Anhang S. 563)
- Shed Work at Periphery (Anhang S. 565)
- Slow it Down (Anhang S. 573).

Ein typischer Anwendungsfall für Mechanismen der Fehlerkompensation ist das Auftreten außergewöhnlich hoher Systemlast, verursacht durch eine ungewöhnlich intensive Nutzung des Systems oder durch mutwillige Störung in Form eines Denial-of-Service-Angriffs (vgl. Anderson 2010:S. 198–199). Die auftretende Systemlast kann hier nicht automatisch reduziert werden. Um die Verfügbarkeit des Systems zu erhalten, ist es deshalb nötig, Maßnahmen zu finden und anzuwenden, die die Auswirkungen des Problems minimieren.

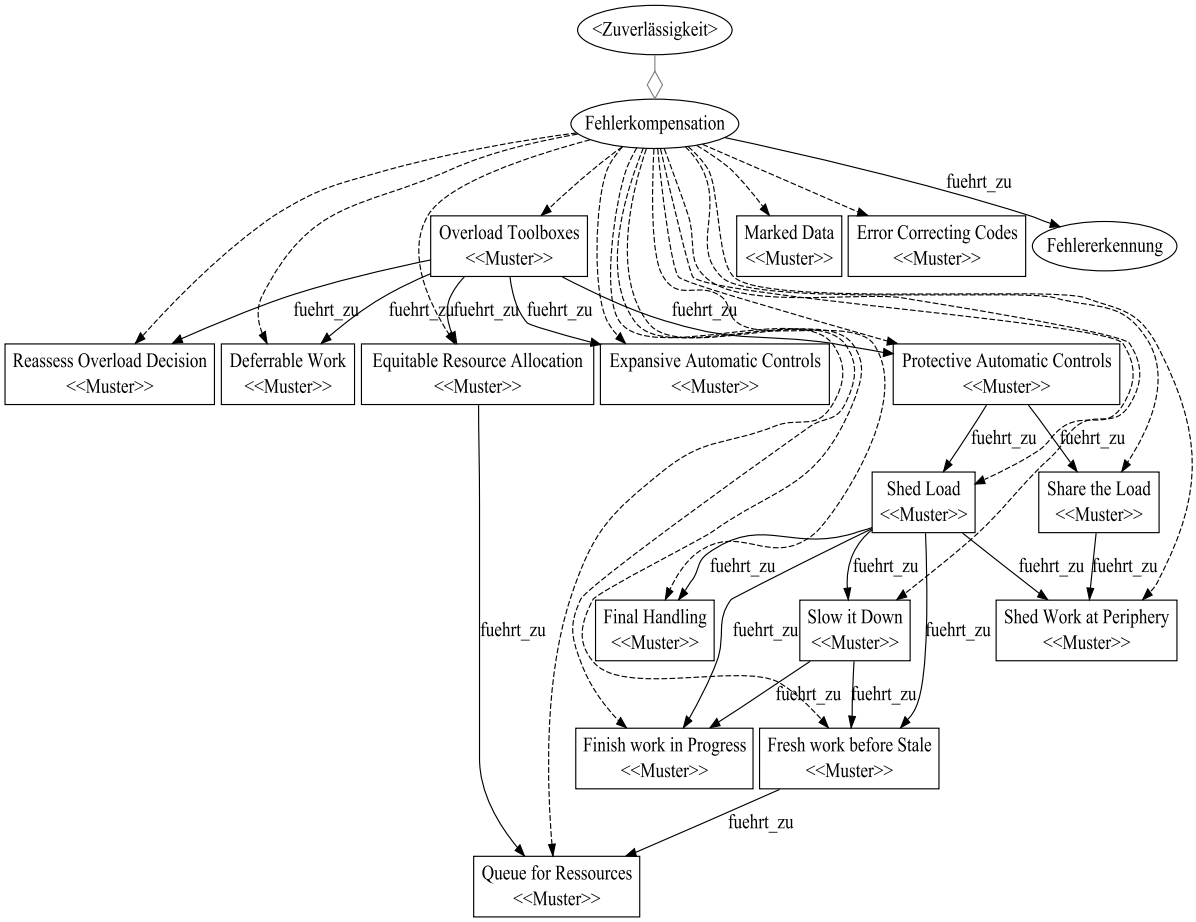



Abbildung 82: Gruppe „Fehlerkompensation“ mit internen Beziehungen

Ein großer Teil der Patterns aus der Gruppe „Fehlerkompensation“ ist geeignet, miteinander kombiniert eine wirkungsvolle Kompensationslösung gegen verschiedene Formen von Überlast-Szenarios aufzubauen. Das Pattern „Reassess Overload Decision“ beschreibt

beispielsweise den kompletten Prozess der Anpassung der Überlast-Kompensationsmechanismen eines Softwaresystems an veränderte Rahmenbedingungen.

Ein häufig verwendeter Ansatz zur Bewältigung hoher Systemlast ist die Parallelisierung von Aufgaben durch die Verteilung auf eine der Last angemessene Zahl von Installationen. Dieses Vorgehen wird durch das Pattern „Share the Load“ beschrieben. Änderungen am Verteilungsszenario können beispielsweise nach einer Anpassung der Überlast-Kompensationsmechanismen erfolgen. Um spontan auftretenden Lastspitzen entgegenzuwirken, empfiehlt das Pattern „Queue for Ressources“ die Verwendung einer festen Zahl von Verarbeitungsprozessen oder Threads, welchen ihre Anfragen über eine Warteschlange zugeteilt werden. Auf diese Weise kann die Verarbeitung der Anfragen, die innerhalb kurzfristiger Lastspitzen die Kapazität des Dienstes übersteigen, auf spätere Phasen mit geringerer Systemlast verschoben werden.

Auch andere Patterns der Gruppe befassen sich mit der Aufgabe, Systemausfälle bei drohender Systemüberlastung zu verhindern. Beispiele dafür sind die Patterns „Shed Load”, „Shed Work at Periphery“, „Slow it Down”, „Fresh Work Before Stale”, „Finish Work in Progress”, „Deferrable Work”, „Queue for Ressources” und „Reassess Overload Decision”.

Die Patterns „Marked Data“ und „Error Correcting Codes“ betrachten einen anderen Aspekt der Fehlerkompensation. Sie befassen sich mit der Kompensation von Fehlern in den Daten. Die möglichen Quellen für solche Datenfehler reichen von technisch bedingten Übertragungsfehlern bis hin zu Eingabefehlern. Das Pattern „Error Correcting Codes“ kommt primär bei technisch bedingten Datenfehlern zum Einsatz. Beispiele hierfür sind Prüfsummen bei der Netzwerkkommunikation via TCP (vgl. Information Sciences Institute University of Southern California 1981:Abschnitt „Reliability“) oder die Fehlerkorrektur in RAID-5-Systemen (vgl. Breitling 2001:S. 30). Das Pattern „Marked Data“ beschreibt die Markierung von fehlerhaften Daten. Es findet Anwendung, wenn eine Korrektur des Fehlers mittels „Error Correcting Codes“ nicht möglich ist und die Auswirkungen des Fehlers eine Behandlung durch Patterns wie „Rollback“ oder „Roll-Forward“ aus der Gruppe der „Recovery Types“ nicht rechtfertigen. Durch die angebrachte Markierung können die fehlerhaften Daten anschließend einer Überprüfung zugeführt oder zum Zweck der Dokumentation weiterverarbeitet werden.

# 6. Entwicklung verteilter Krankenakten

Die Entwicklung verteilter Krankenakten ist die Entwicklung komplexer verteilter Systeme im Anwendungsbereich medizinischer Datenverarbeitung. Sie setzt ein systematisches und problemadäquates Vorgehen zur Konzeption des zugehörigen Softwaresystems voraus. Die Kapitel 3, 4 und 5 beschreiben das Konzept und die Bestandteile einer Methode zur Entwicklung solcher Systeme unter Zuhilfenahme von Patterns. Der Hauptbestandteil dieser Methode ist der in Kapitel 5 beschriebene Kernbestandteil einer Pattern-Sprache für verteilte Krankenakten. Er ist intern nach dem in Kapitel 4 beschriebenen Konzept strukturiert und genügt, wie in Kapitel 3 dargestellt, den in Kapitel 2 ermittelten Anforderungen an verteilte Krankenakten.

Die Entwicklung von verteilten Krankenakten selbst beinhaltet die detaillierte Analyse des konkreten Anwendungsgebiets der Akte und reicht von der Konzeption der zugehörigen Architektur bis hin zur Umsetzung des Softwaresystems in einer konkreten Programmiersprache sowie für eine konkrete Infrastruktur. In diesem Kapitel wird die Vorgehensweise zur schrittweisen Entwicklung einer verteilten Krankenakte entsprechend der vorgestellten Methodik erklärt. Das geschieht sowohl allgemein, als auch am Beispiel eines praktisch umgesetzten und im Einsatz befindlichen Softwaresystems.

## *6.1. Anforderungsbezogene Patternauswahl*

Der Kernbestandteil der Pattern-Sprache für verteilte Krankenakten enthält einen umfangreichen Pattern-Katalog, dessen Patterns Lösungen zu vielfältigen Problemen bieten. Für die Entwicklung eines konkreten Softwaresystems sind nicht alle Patterns dieses Katalogs relevant. Welche Patterns das sind, ist abhängig von den Anforderungen, die für die konkret zu entwickelnde verteilte Krankenakte gelten.

Grundsätzlich ist es die Aufgabe einer Pattern-Sprache, die problembezogene Auswahl von Patterns durch die eigene Struktur und dokumentierte Auswahlverfahren zu unterstützen. Beck und Cunningham erläutern in einer ihrer frühen Publikationen zur Verwendung von Pattern-Sprachen im Software-Engineering (Kent Beck & Cunningham 1987), dass eine Pattern-Sprache den Entwickler durch das Anbieten umsetzbarer Lösungen für alle bekannten Probleme, die im Laufe der Entwicklung einer Software auftreten können, führen muss. Buschmann, Henney und Schmidt (Buschmann u. a. 2007b:S. 327) verfeinern diese Aussage und schränken sie auf die Anwendungsdomäne der Pattern-Sprache mit ihren definierten Aufgabenschwerpunkten und Problembereichen ein. Sie fordern, dass eine Pattern-Sprache einen oder mehrere definierte Einstiegspunkte besitzt und dass definierte Pfade für die Patternauswahl existieren.

Kneubühl u. a. (Kneubühl u. a. 2009) untersuchen in ihrer Masterarbeit Methoden zur Auswahl von Patterns für die Gestaltung von Benutzerschnittstellen. Ihre Ergebnisse sind aufgrund des unterschiedlichen Anwendungsgebietes nicht in vollem Umfang auf verteilte Krankenakten übertragbar. Eine ihrer im Rahmen der Anforderungsanalyse getroffenen Aussagen (Kneubühl u. a. 2009:S. 18) ist auch auf die hier vorliegende Pattern-Sprache anwendbar. Sie beschreibt dass es  Notwendig ist, dass sich die Patternauswahl in den Ablauf

des Entwicklungsprozesses der Software integriert und diesen begleitet. Die genannten Anforderungen zur Erleichterung der Findung und Auswahl geeigneter Patterns sind im Strukturkonzept und der Beschreibung des Kernbestandteils der Pattern-Sprache berücksichtigt und dadurch Bestandteile des Konzepts zur anforderungsbezogenen Patternauswahl.

Eine gezielte Verwendung des vorgestellten Pattern-Systems setzt die Kenntnis seines Strukturkonzepts voraus. Dieses ist in Kapitel 4 ausführlich beschrieben und beinhaltet sowohl die Definitionen der enthaltenen Elementtypen als auch ihre grafische Darstellung innerhalb der für die Repräsentation der ermittelten Beziehungen gewählten grafischen Sprache. Die Graphen ermöglichen eine übersichtliche Darstellung der Abhängigkeiten oder Beziehungen zwischen Patterns und unterstützen so die Auswahl verwandter, ergänzender oder benötigter Patterns während der Planung und Entwicklung einer verteilten Krankenakte.

Die textuelle Darstellung des Kernbestandteils der Pattern-Sprache in Kapitel 5 erläutert den Zweck der identifizierten Gruppen sowie die Anwendung einzelner Patterns und Patternsequenzen. Sie dient der Beschreibung der Struktur und der Zusammenhänge innerhalb des Kernbestandteils der Pattern-Sprache für verteilte Krankenakten und vermittelt dem Leser einen Überblick über dessen Inhalt und Verwendung. Eine umfassende Darstellung aller im Rahmen dieser Arbeit identifizierten Beziehungen zwischen den untersuchten Patterns befindet sich in Anhang A. Er umfasst keine vollständige Pattern-Sprache, aber eine vollständige Darstellung der in dieser Arbeit identifizierten Bestandteile dieser Pattern-Sprache. Die Verwendung von Anhang A setzt die Kenntnis der Beschreibung aus Kapitel 5 voraus. Sie ist somit der Ausgangspunkt für eine systematische Patternauswahl innerhalb der Language.

Die Durchführung einer systematischen Patternauswahl für eine konkrete verteilte Krankenakte erfordert, dass im Vorfeld das fachliche Aufgabengebiet und das Einsatzgebiet festgelegt sind. Ist diese Vorbedingung erfüllt, kann ein systematischer Prozess der Patternauswahl durchgeführt werden. Er beginnt mit der Phase der Anforderungsanalyse. Die dazu vorgesehenen Gruppen und Patterns sind für die vier Schwerpunkte *Flexibilität*, *Kommunikation*, *Sicherheit* und *Zuverlässigkeit* in den Kapiteln 5.1.1, 5.2.1, 5.3.1 und 5.4.1 beschrieben. Die in den Kapiteln abgebildeten Pattern-Diagramme der Gruppen visualisieren durch die abgebildeten Beziehungen die Reihenfolge der Verwendung der Gruppen und Patterns innerhalb des jeweiligen Schwerpunkts. In der Ebene *Anforderungsanalyse* können die Schwerpunkte in beliebiger Reihenfolge bearbeitet werden, da zwischen den enthaltenen Gruppen und Patterns keine Abhängigkeiten bestehen, die zu rückwirkenden Änderungen in den Ergebnissen der vorher genutzten Patterns anderer Schwerpunkte führen. Das Ergebnis der Phase *Anforderungsanalyse* ist eine Sammlung detailliert spezifizierter Anforderungen an die Softwarearchitektur und das Softwaredesign der zu entwickelnden verteilten Krankenakte.

Die nächste Phase der Patternauswahl ist die Gestaltung der Softwarearchitektur. Der Übergang in diese Ebene erfolgt über die ausgehenden Beziehungen der Anforderungspatterns aus der Ebene *Anforderungsanalyse*. Sie vermitteln in Abhängigkeit von den Ergebnissen der Anforderungsanalyse, den Anforderungen, zu Architekturpatterns oder Gruppen von Architekturpatterns, die geeignet sind, die entsprechenden Anforderungen zu befriedigen. Jede dieser Beziehungen entspricht einer kontextabhängig zu betrachtenden Kombinations-

empfehlung. Die Entscheidung über die tatsächliche Verwendung dieser Empfehlung liegt im Ermessen des Softwarearchitekten oder Entwicklers. Manche Gruppen der Ebene *Architektur* werden nicht direkt von Anforderungspatterns aus referenziert. Bei diesen Gruppen sind die enthaltenen Patterns entweder individuell von unterschiedlichen Anforderungen abhängig oder werden indirekt durch die Beziehungen direkt referenzierter Patterns in Beziehung zu Anforderungen gesetzt. Die für diese Phase der Auswahl notwendige Dokumentation der Patterns und Beziehungen befindet sich in Anhang A. Darin sind alle im Kernbestandteil der Pattern-Sprache für verteilte Krankenakten enthaltenen Patterns mit ihren Beziehungen, einschließlich ihrer von Gruppen geerbten Beziehungen dokumentiert. Sie ermöglichen eine schrittweise und anforderungsabhängige Auswahl geeigneter Patterns. Die gleiche Vorgehensweise wird auch für den Übergang zur Design-Ebene sowie innerhalb der Ebene *Design* selbst angewendet.

Besonderheiten, die sich auf die Abhängigkeiten innerhalb einzelner Gruppen beziehen, sind in der Beschreibung der entsprechenden Gruppe in Kapitel 5 dokumentiert und müssen im Rahmen der Patternauswahl berücksichtigt werden. Ein Beispiel für eine solche Besonderheit ist der beim Pattern „Units of Mitigation“ (siehe S. 191) genannte Rückkopplungseffekt, der eine nochmalige Anforderungsanalyse, bezogen auf kleinere Einheiten, die durch die Anwendung von Architekturpatterns entstanden sind, fordert.

Der beschriebene Auswahlprozess gibt die initiale Lösungsauswahl hinsichtlich der Konzeption der Softwarearchitektur und des Softwaredesigns wieder. Neben der initialen Lösungsauswahl ist auch die sukzessive Verbesserung der geplanten Architektur und des geplanten Designs Bestandteil der Entwicklung verteilter Krankenakten. Für diese Aufgaben im Entwicklungsprozess sind individuell angepasste Vorgehensweisen angebracht.

Bei einer vom systematischen Auswahlprozess abweichenden Nutzung der Pattern-Sammlung, beispielsweise zum Zeitpunkt eines Reviews oder der problembezogenen Nachbesserung bereits erstellter Modelle kann alternativ eine direkte Patternauswahl durchgeführt werden. Sie führt entweder über Kapitel 5 und die darin vorgestellten problemorientierten Gruppen oder über die Abhängigkeiten bereits verwendeter Patterns zur Auswahl von einem oder mehreren Patterns zur Behebung des identifizierten Problems. Wird z. B. in einer Komponente des geplanten Systems einer verteilten Krankenakte ein Flexibilitätsdefizit festgestellt, so kann innerhalb des Schwerpunkts *Flexibilität* nach einer zu dem Problem passenden Gruppe gesucht und darin ein geeigneter Lösungsansatz gefunden werden.

## 6.2. Einweiserportal am Universitätsklinikum Regensburg

Die Entwicklung von Softwaresystemen für verteilten Krankenakten umfasst nach der vorliegenden Entwicklungsmethode die Auswahl und Anwendung einer Menge von Patterns aus einer speziell für diesen Zweck konzipierten Patternsammlung. Eine abstrakte Beschreibung des Auswahlverfahrens ist durch die anforderungsbezogene Patternauswahl definiert. In den Kapiteln 6.3 und 6.4 ist durch die Anwendung der anforderungsbezogenen Patternauswahl auf eine praktische Aufgabenstellung aus dem Bereich verteilter Krankenakten beispielhaft dargestellt, wie das Konzept der Patternauswahl zu verwenden ist.

Dadurch wird zugleich auch die Verwendbarkeit sowohl der beschriebenen Pattern-Sammlung als auch der Methode zur Patternauswahl belegt.

Als praktisches Beispiel dient das Einweiserportal des Universitätsklinikums Regensburg. Es bietet Funktionen zur Bereitstellung und zum Abruf von medizinischer Dokumentation, die am Universitätsklinikum Regensburg vorliegt. Zugriffsberechtigte Einweiser aus Einrichtungen außerhalb des Universitätsklinikums können darüber auf Befunde und sonstige Dokumente zu den von ihnen eingewiesenen Patienten zugreifen. Das Einweiserportal wurde zwischen 2007 und 2009 im Rahmen eines protptypischen Pilotprojekts für die medizinische Klinik II des Universitätsklinikums Regensburg entwickelt. Die zugriffsberechtigten Einweiser sind seit Beginn des Projekts ausgewählte niedergelassene Kardiologen und zuweisende Kreiskrankenhäuser aus den Landkreisen Regensburg, Kelheim und Straubing.

Das Einweiserportal des Universitätsklinikums Regensburg besitzt eine vollständig in die Arztbrief- und Befundschreibung integrierte Freigabe von Dokumenten und bietet dem externen Benutzer via Webbrowser Zugriff auf freigegebene Inhalte. Einige der Lösungsansätze und Eigenschaften zur transparenten Integration in die Systemlandschaft des Universitätsklinikums Regensburg sind in Form der drei Publikationen *„Generischer Architekturansatz für Telemedizinportale und verteilte Krankenakten“* (Wiedermann u. a. 2008), *„Service oriented approach for multi backend retrieval in medical systems“* (Wiedermann u. a. 2009) und *„Vereinfachung von DICOM-Querys durch eine domänenspezifische Abfragesprache“* (Kraus u. a. 2008) beschrieben.

Webportale zur Bereitstellung oder zum Austausch von Befunddokumenten sind in der Literatur häufig beschrieben. Im Rahmen der Entwicklung des Einweiserportals sind neben den verwendeten Patterns auch Ideen, entstanden durch die Betrachtung der Vor- und Nachteile bestehender und publizierter Befundportale, eingeflossen. Insbesondere waren das das Telemedizinportal TempoBy des Klinikums München Großhadern, das Produkt „Soarian Integrated Care“ von Siemens sowie die Quellen (Pschichholz u. a. 2008; Kuchenbecker & Behrens-Baumann 2004; Bergmann u. a. 2005; Kammerer u. a. 2009; Duennebeil u. a. 2009).

Aus der Analyse der angegebenen Systeme und Quellen resultiert eine Sammlung von Anforderungen, die seit der ersten Entwicklungsphase die Gestaltung des Einweiserportalsystems prägen. Dabei handelt es sich um die folgenden Anforderungen:

- Zur Bereitstellung von Daten für externe Benutzer ist kein manuelles Kopieren in das Einweiserportal notwendig.
- Die Freigabe von Daten ist direkt in den Prozess der medizinischen Dokumentation zu integrieren.
- Es darf keine technologischen Brüche zwischen dem am Universitätsklinikum Regensburg bestehenden Krankenhausinformationssystem und dem Einweiserportal geben.

## 6.3. Anforderungsanalyse

Die erste Phase bei der Entwicklung verteilter Krankenakten ist die Phase der Analyse und Definition der Anforderungen. Der Kernbestandteil der Pattern-Sprache enthält hierfür Patterns in allen vier Schwerpunkten. Zur Verbesserung der Übersichtlichkeit werden nachfolgend die Analyseschritte und Ergebnisse zu jeweils zwei Schwerpunkten gemeinsam bearbeitet und dargestellt. Dazu werden die Schwerpunkte *Kommunikation* und *Flexibilität* sowie *Sicherheit* und *Zuverlässigkeit* jeweils zusammenhängend betrachtet. Dieses Vorgehen lässt sich damit begründen, dass in der Anforderungsanalyse eine enge Kopplung zwischen den Schwerpunkten *Kommunikation* und *Flexibilität* vorliegt. Bei den Schwerpunkten *Sicherheit* und *Zuverlässigkeit* liegt eine enge Kopplung im Bereich der Anforderungsdefinition vor. Der Grad der Kopplung definiert sich dabei aus der Menge der Patterns, die in den entsprechenden Phasen in beiden Schwerpunkten gemeinsam genutzt werden.

### 6.3.1. Rahmenbedingungen für Kommunikation und Flexibilität

Die erste Aufgabe der Anforderungsanalyse umfasst die Ermittlung und Analyse der Rahmenbedingungen für die Gestaltung der Kommunikation zwischen den Bestandteilen der verteilten Krankenakte und die Ermittlung der Bereiche, in denen eine erhöhte Wahrscheinlichkeit von Veränderung besteht. Sie umfasst die Patterns aus den Gruppen „Analyse des Kommunikationsbedarfs“ und „Analyse flexibilitätsbeeinflussender Faktoren“, den Einstiegsgruppen in die Schwerpunkte *Kommunikation* und *Flexibilität*. Zur Durchführung der Analyse der Rahmenbedingungen von Kommunikation und Flexibilität sind die Patterns

- Analyse der organisatorischen Verteilung im Verbund (Anhang S. 272),
- Analyse der betroffenen Behandlungspfade (Anhang S. 267),
- Untersuchung betroffener Subsysteme (Anhang S. 595) und
- Veränderlichkeitsanalyse (Anhang S. 599)

in der angegebenen Reihenfolgen anzuwenden. Die „Analyse der organisatorischen Verteilung im Verbund“ ergibt im Schritt 1 eine Liste portalrelevanter Einrichtungen, bestehend aus der Medizinischen Klinik II des Universitätsklinikums Regensburg sowie zwei Kreiskrankenhäusern und zwei Praxen niedergelassener Kardiologen. Im Schritt 2 ist die Liste aus Schritt 1 um das für die verteilte Krankenakte relevante Leistungsspektrum der jeweiligen Einrichtung zu ergänzen. Da ausschließlich das Universitätsklinikum Regensburg Inhalte über das Einweiserportal bereitstellen möchte, sind die beiden Kreiskrankenhäuser und die niedergelassenen Kardiologen nur als konsumierende Benutzer des Einweiserportals zu betrachten. Das portalrelevante Leistungsspektrum der Medizinischen Klinik II umfasst die Leistungen des Herzkatheterlabors, die Echokardiografie sowie EKG und Lungenfunktion. Schritt 3 der Analyse der organisatorischen Verteilung im Verbund beinhaltet die Klassifikation der beteiligten Einrichtungen nach ihrer finanziellen und IT-technischen Leistungsfähigkeit. Von den beteiligten Einrichtungen besitzt ausschließlich das Universitätsklinikum umfassende finanzielle und technische Mittel für die Realisierung und den Betrieb komplexer Softwaresysteme.

Als nächstes Pattern folgt die „Analyse der betroffenen Behandlungspfade“. Es ist wie die „Analyse der organisatorischen Verteilung im Verbund“ eine Vorbedingung für die „Untersuchung der betroffenen Subsysteme“ (siehe S. 595) und hat selbst die „Analyse der organisatorischen Verteilung im Verbund“ als notwendige Voraussetzung (siehe S. 267). Da das Konzept des Einweiserportals keine Datenerfassung durch externe Einrichtungen vorsieht, bildet es auch keine vollständigen einrichtungsübergreifenden Behandlungspfade ab. Weitere Erkenntnisse, die aus einer Betrachtung der zugehörigen klinikinternen Pfade gezogen werden können, sind bereits in Kapitel 2.3 umfassend behandelt worden. Aus diesem Grund wird zur Vereinfachung nur Schritt 1 des Patterns „Analyse der betroffenen Behandlungspfade“ angewandt. Er beinhaltet die Erfassung einer Liste von für das Portal relevanten Leistungen. Für die konkrete Planung des Einweiserportals wird diese Liste um die jeweils aus den Leistungen resultierenden Dokumente erweitert. Es handelt sich um die Dokumente *Arztbrief*, *Herzkatheterbefund*, *Herzkatheterfilm*, *Echokardiografie-Befund*, *EKG-Kurve* und *Lungenfunktionskurve*. Sie sollen über das Einweiserportal berechtigten Benutzern zur Einsicht zur Verfügung gestellt werden.

Der nächste Schritt der Analyse der Rahmenbedingungen für die Anforderungsdefinition bezüglich der Aspekte *Kommunikation* und *Flexibilität* ist die „Untersuchung der betroffenen Subsysteme“. Sie gründet auf der Tatsache, dass es derzeit kaum möglich ist, ein medizinisches Softwaresystem zu entwickeln, das nicht in eine Umgebung mit bereits existierenden medizinischen Softwaresystemen zu integrieren ist. Für die Ermittlung der relevanten Softwaresysteme wird vorausgesetzt, dass bereits alle beteiligten Einrichtungen mit den zugehörigen relevanten Inhalten erkannt und festgehalten sind. Dadurch können, ausgehend von den Inhalten bzw. Dokumenten die Systeme ermittelt werden, in denen diese Dokumente verwaltet und gespeichert sind. Bezogen auf das Einweiserportal ergibt sich folgendes Ergebnis. Die Dokumente *Arztbrief*, *Echokardiografie-Befund* und *Herzkatheterbefund* werden im Krankenhausinformationssystem SAP is.h.med verwaltet. Für *Herzkatheterfilme* existiert ein eigenes PACS. *Lungenfunktions-* und *EKG-Kurven* werden in einem Dokumentenarchiv der Firma Easy gespeichert und durch Anforderungseinträge in SAP is.h.med referenziert.

Der Grad an Flexibilität, der in einem Bestandteil des Softwaresystems des Einweiserportals erreicht werden muss, wird durch die Wahrscheinlichkeit von Änderungen in diesem Bereich bestimmt. Aus diesem Grund ist die Veränderlichkeitsanalyse der Ausgangspunkt für die Ermittlung von Anforderungen, die die Flexibilität des Softwaresystems beeinflussen. Sie basiert auf den Ergebnissen der Patterns „Untersuchung betroffener Subsysteme“ und „Analyse der betroffenen Behandlungspfade“. Subsysteme, Pfade und Inhalte werden dabei anhand ihrer Änderungswahrscheinlichkeit bewertet. Außerdem werden allgemeine Veränderungsrisiken, die das zu entwickelnde System selbst betreffen, berücksichtigt. Für das Einweiserportal sind die folgenden zu erwartenden Veränderungen festzustellen:

V1: Erweiterung des Portals um zusätzliche externe Einrichtungen und Nutzer
V2: Existierende IT-Systeme am Universitätsklinikum werden durch neuere oder andere Software ersetzt
V3: Zusätzliche Inhalte sollen über das Einweiserportal verfügbar gemacht werden
V4: Verbindungsparameter zu den betroffenen Subsystemen ändern sich.

Die zu erwartende Veränderung V1 ist häufig, die anderen Veränderungen sind gelegentlich zu erwarten. Keine der Veränderungen ist als unwahrscheinlich oder selten zu klassifizieren. Daher muss das Einweiserportal bezüglich aller vier Veränderungen flexibel gestaltet werden.

### 6.3.2. Anforderungen an Kommunikation und Flexibilität

Aufbauend auf den Rahmenbedingungen für die Kommunikation und die Flexibilität können die konkreten Anforderungen an die Kommunikationsinfrastruktur und die Flexibilität der Softwarearchitektur sowie das Softwaredesign des Einweiserportals definiert werden. Nach dem Ausschluss von Patterns aufgrund verschiedener Gründe verbleibt die folgende Liste als Pattern-Katalog für die Anforderungsanalyse der Schwerpunkte *Kommunikation* und *Flexibilität* des Einweiserportals des Universitätsklinikums Regensburg:

- Inter System Interface Requirements (Anhang S. 412)
- Technology Requirements (Anhang S. 585)
- Scalability Requirements (Anhang S. 541)
- Response Time Requirements (Anhang S. 516)
- Extendability Requirements (Anhang S. 370)
- Installability Requirements (Anhang S. 407)
- Unparochialness Requirements (Anhang S. 593).

Das Pattern „Inter System Interaction Requirements“ ist aufgrund der über die Menge der Schnittstellen des Einweiserportals hinweg gleichförmigen Interaktion nicht aufgeführt. Das Pattern „Process Adaptivity Requirements“ ist ausgeschlossen, da keine einrichtungsübergreifenden Prozesse abgebildet werden. Das Pattern „Multi-Lingual Requirements“ wird nicht verwendet, weil der Einzugsbereich des Einweiserportals keine Mehrsprachigkeit erforderlich macht. „Multiness Requirements“ werden nicht explizit erfasst, da alle relevanten „Multiness Requirements“ gleichzeitig als „Extendability Requirements“ zu erfassen sind und der Zweck der „Extendability Requiremens“ dem Wesen der Anforderungen besser gerecht wird. Dieses Vorgehen zeigt das als anforderungsbezogene Patternauswahl beschriebene gezielte und problembezogene Wählen und Verwenden der verfügbaren Patterns.

Das erste Pattern der Anforderungsdefinition beüglich der Kommunikation innerhalb des Einweiserportalsystems ist das Pattern „Inter System Interface Requirements“. Es basiert auf den Ergebnissen der Patterns „Analyse der organisatorischen Verteilung im Verbund“ und „Untersuchung der betroffenen Subsysteme“. Außerdem kann sein Ergebnis von notwendigen „Comply-with Standard Requirements“ und bereits getroffenen Architekturentscheidungen bezüglich der Verteilung der Dokumente pro Patient innerhalb des Gesamtsystems abhängen. In seiner Darstellung umfasst es ein System-Kontext-Diagramm (siehe Abbildung 83) sowie eine detaillierte Beschreibung der darin dargestellten Schnittstellen.

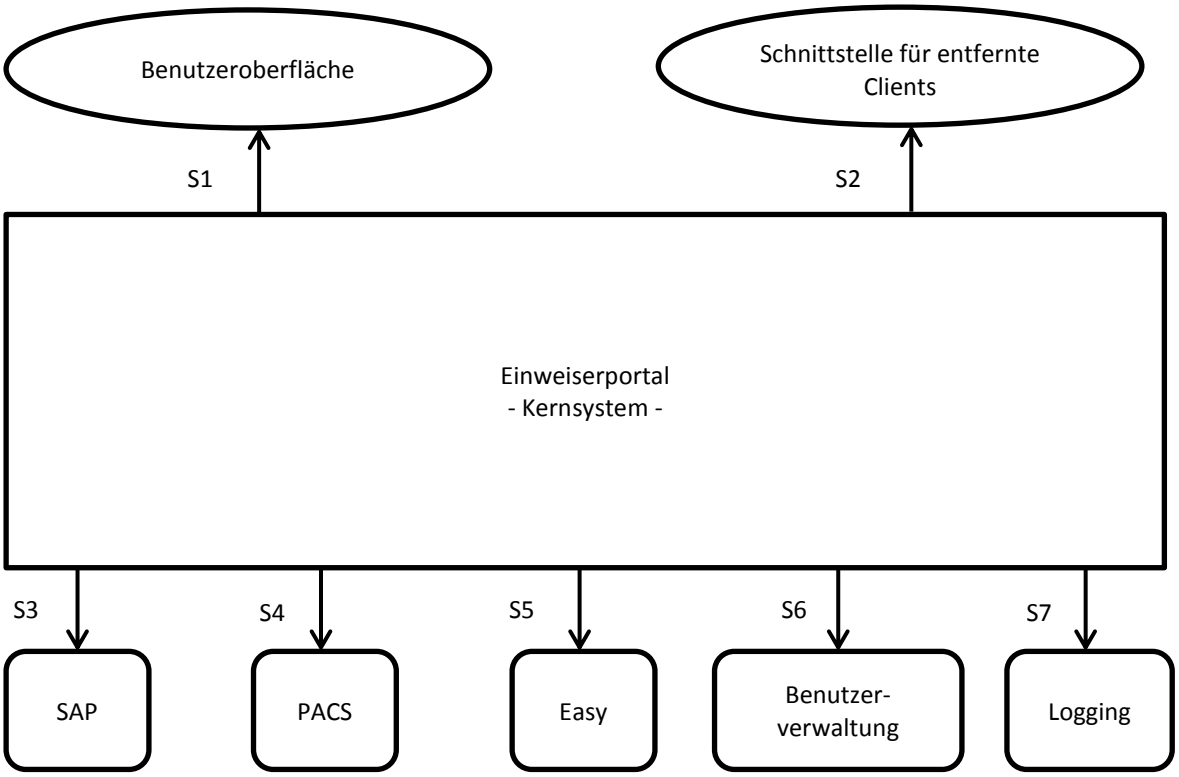



Abbildung 83: System-Kontext-Diagramm zum Einweiserportal

Das System-Kontext-Diagramm stellt das Einweiserportal mit seinen Schnittstellen zu anderen Systemen dar. Dabei sind die Schnittstellen S1 und S2 Bestandteil des Portalsystems und können bei der Entwicklung frei gestaltet werden. Sie sind folglich keine klassischen Inter System Interfaces und werden deshalb bei der Anwendung des Patterns nicht mit spezifiziert. Für die Schnittstellen S3 bis S5 gilt, dass sie die Verbindung zu bereits existierenden und explizit als Kommunikationspartner ausgewählten Systemen abbilden. Sie stehen somit im Fokus der Betrachtung der „Inter System Interface Requirements“.

Die Schnittstellen S6 und S7 können sowohl zur Anbindung eigener Implementierungen als auch zur Anbindung externer Dienste an das Einweiserportal genutzt werden. Sie erfordern einen noch höheren Grad an Flexibilität als die Schnittstellen S3 bis S5. Um die Menge der Schnittstellendefinitionen übersichtlich zu halten, werden im nachfolgenden Text ausschließlich die Schnittstellen S3 bis S5 dargestellt, da sie am meisten den Kernaspekten des Einweiserportals entsprechen. Sollte eine der Schnittstellen aus mehreren logischen Bestandteilen bestehen, so werden diese durch eine nachgestellte, mit Dezimalpunkt abgetrennte Nummerierung in der Schnittstellennummer kenntlich gemacht.

| ***Schnittstellenname*** | ***Patientensuche*** |
|---|---|
| *Schnittstellennummer* | *S3.1* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: SAP IS.H.MED* |
| *Zweck der Schnittstelle* | *Benutzerrechteabhängige Patientensuche* |
| *Schnittstelleneigner* | *Einweiserportal-Projekt* |
| *Standard* | *Keiner* |
| *Technologie* | *SAP RFC in Verbindung mit ABAP-Funktionsbausteinen und dem SAP Java Connector (JCo)* |

| | |
|---|---|
| ***Schnittstellenname*** | ***Arztbrief*** |
| *Schnittstellennummer* | *S3.2* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: SAP IS.H.MED* |
| *Zweck der Schnittstelle* | *a) Ermitteln einer Liste von Arztbriefen zu einem Patienten in Abhängigkeit der Berechtigungen des angemeldeten Benutzers*<br>*b) Übermitteln eines konkret ausgewählten Arztbriefes als PDF-Dokument.* |
| *Schnittstelleneigner* | *Einweiserportal-Projekt* |
| *Standard* | *Keiner* |
| *Technologie* | *SAP RFC in Verbindung mit ABAP-Funktionsbausteinen und dem JCo* |

| | |
|---|---|
| ***Schnittstellenname*** | ***Herzkatheterbefund*** |
| *Schnittstellennummer* | *S3.3* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: SAP IS.H.MED* |
| *Zweck der Schnittstelle* | *a) Ermitteln einer Liste von Herzkatheterbefunden zu einem Patienten in Abhängigkeit der Berechtigungen des angemeldeten Benutzers*<br>*b) Übermitteln eines konkret ausgewählten Herzkatheterbefunds als PDF-Dokument.* |
| *Schnittstelleneigner* | *Einweiserportal-Projekt* |
| *Standard* | *Keiner* |
| *Technologie* | *SAP RFC in Verbindung mit ABAP-Funktionsbausteinen und dem JCo* |

| | |
|---|---|
| ***Schnittstellenname*** | ***EKG-Patienten-Zuordnung*** |
| *Schnittstellennummer* | *S3.4* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: SAP IS.H.MED* |
| *Zweck der Schnittstelle* | *Ermitteln einer Liste von EKG-Kurven und Befunden zu einem Patienten einschließlich der Referenz zu diesen Kurven und Befunden im Easy-Archiv (vgl. S5.1)* |
| *Schnittstelleneigner* | *Einweiserportal-Projekt* |
| *Standard* | *Keiner* |
| *Technologie* | *SAP RFC in Verbindung mit ABAP-Funktionsbausteinen und dem JCo* |

| | |
|---|---|
| ***Schnittstellenname*** | ***LUFU-Patienten-Zuordnung*** |
| *Schnittstellennummer* | *S3.5* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: SAP IS.H.MED* |
| *Zweck der Schnittstelle* | *Ermitteln einer Liste von LUFU-Kurven und Befunden zu einem Patienten einschließlich der Referenz zu diesen Kurven und* |

| | |
|---|---|
| | *Befunden im Easy-Archiv (vgl. S5.2)* |
| *Schnittstelleneigner* | *Einweiserportal-Projekt* |
| *Standard* | *Keiner* |
| *Technologie* | *SAP RFC in Verbindung mit ABAP-Funktionsbausteinen und dem JCo* |

| | |
|---|---|
| ***Schnittstellenname*** | ***Herzkatheterfilm*** |
| *Schnittstellennummer* | *S4.1* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: Herzkatheter-PACS* |
| *Zweck der Schnittstelle* | *Laden eines Herzkatheterfilms aus dem Bildarchiv zur Verarbeitung und Anzeige im Einweiserportal* |
| *Schnittstelleneigner* | *Philips DECOM* |
| *Standard* | *DICOM* |
| *Technologie* | *Java API DQL (Dicom Query Language) – vgl.* (Kraus 2008; Kraus u. a. 2008) |

| | |
|---|---|
| ***Schnittstellenname*** | ***EKG-Kurve*** |
| *Schnittstellennummer* | *S5.1* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: Easy Dokumentenarchiv* |
| *Zweck der Schnittstelle* | *Laden einer EKG-Kurve als PDF für die Anzeige im Einweiserportal* |
| *Schnittstelleneigner* | *Easy* |
| *Standard* | *Keiner* |
| *Technologie* | *XML-RPC mit vom Hersteller Easy definierten XML-Request- und Response-Nachrichten.* |

| | |
|---|---|
| ***Schnittstellenname*** | ***LUFU-Kurve*** |
| *Schnittstellennummer* | *S5.2* |
| *Beteiligte Systeme* | *Client: Einweiserportal*<br>*Server: Easy Dokumentenarchiv* |
| *Zweck der Schnittstelle* | *Laden einer LUFU-Kurve als PDF für die Anzeige im Einweiserportal* |
| *Schnittstelleneigner* | *Easy* |
| *Standard* | *Keiner* |
| *Technologie* | *XML-RPC mit vom Hersteller Easy definierten XML-Request- und Response-Nachrichten.* |

Der nächste Schritt ist die Definition der „Technology Requirements“. Sie dienen der Festlegung von Anforderungen bezüglich der Technologien, die bei der Umsetzung des Einweiserportals in bestimmten Bereichen verwendet werden müssen. Das zugehörige Pattern dient nicht dazu, Technologie-Entscheidungen zu treffen (vgl. Withall 2007:S. 65). Für jede der definierten Anforderungen muss somit ein zwingender Grund vorliegen, der den Einsatz der angegebenen Technologie erforderlich macht. Zwischen dem Pattern „Inter System Interface Requirements“ und dem Pattern „Technology Requirements“ existiert eine Führt-zu-

Beziehung. Sie gründet auf der Tatsache, dass es hilfreich ist, bei der Spezifikation von technologischen Anforderungen auf eine Auflistung der relevanten Technologien aus den im System einzuplanenden Schnittstellen zurückgreifen zu können.

Im Bereich der Anwender des Einweiserportals ist, wie bereits durch Schritt 3 der Analyse der organisatorischen Verteilung im Verbund festgestellt, keine besondere IT-Kompetenz vorauszusetzen. Eine jeweils lokal zu installierende Clientanwendung sorgt für zusätzliche Komplexität und zusätzlichen Administrationsaufwand auf Rechnern der externen Benutzer. Außerdem unterstützt die am Universitätsklinikum verfügbare VPN-Gateway primär webbasierten Zugriff. Aus diesen Gründen ist als Technologie für die Benutzeroberfläche eine webbasierte Oberfläche zu wählen.

R1.1: Webbasierte Benutzeroberfläche

Einige der Systeme (siehe „Inter System Interface Requirements“), zu denen Schnittstellen bestehen müssen, sind SAP-R/3-Systeme. Erweiterungen, die innerhalb dieser Systeme zu schreiben sind, müssen daher in ABAP, der Programmiersprache von SAP, verfasst werden.

R1.2: Schnittstellenprogramme zu SAP in ABAP programmieren

SAP bietet mit dem Java Connector JCo[20] eine geeignete Lösung zur Kommunikation zwischen Java- und ABAP-Programmen über das proprietäre Kommunikationsprotokoll SAP-RFC. Eine vergleichbare Lösung ist auch für .NET verfügbar. Eine der noch nicht spezifizierten, aber gewünschten Anforderungen an das Einweiserportal besagt, dass das System selbst Plattform- und Hersteller-neutral entwickelt werden soll. Das, sowie die gute Verfügbarkeit von API für die Anbindung von DICOM-fähigen Bildarchiven und des Easy-Archivs machen Java zur geeignetsten Entwicklungsplattform.

R1.3: Kernsystem in Java
R1.4: Kommunikation zu SAP IS.H.MED mittels JCo

Der nächste Schritt ist die Definition der „Scalability Requirements“ (Skalierbarkeits-anforderungen). Sie definieren, in welcher Richtung, welcher Form und welchem Umfang das Einweiserportal in der Lage sein muss, zu wachsen, ohne dass das Softwaresystem selbst verändert werden muss. Für das Einweiserportal ist Skalierbarkeit bezüglich der Anzahl der Kommunikationspartner bzw. externen Benutzer, der Anzahl von Anfragen pro Zeiteinheit und der Anzahl der integrierten Datenanbieter bzw. Dokumentenarten relevant.

Das System soll geeignet sein, mit der Zahl der beteiligten medizinischen Einrichtungen und der daraus resultierenden Zahl an Benutzern wachsen zu können, da zum Zeitpunkt der Planung des Systems sein endgültiger Verbreitungsgrad nicht exakt bekannt ist. Deshalb soll das Einweiserportal von wenigen Benutzern und Patienten bis hin zu einigen tausend Benutzern und mehreren Millionen Patienten ohne Modifikation des Programmcodes skalierbar sein.

[20] Vgl. http://help.sap.com/saphelp_nwpi711/helpdata/de/48/70792c872c1b5ae10000000a42189c/frameset.htm

R2.1: Muss mehrere Tausend Benutzer und mehrere Millionen Patienten unterstützen

Um auch große Mengen von Anfragen kompensieren zu können, soll das System so gestaltet sein, dass es auf mehr als einen Rechner zur gleichzeitigen Bearbeitung von Anfragen verteilt werden kann. Eine solche Verteilung muss möglich sein, darf aber nicht notwendig sein, um in Einsatzszenarien mit geringer Anfragehäufigkeit nicht unnötige Kosten zu verursachen und Ressourcen zu binden.

R2.2: Muss parallel auf mehreren Rechnern betrieben werden können

Außerdem ist es möglich, dass weitere Organisationseinheiten des Universitätsklinikums über das Einweiserportal Inhalte für externe Benutzer zur Verfügung stellen wollen. Dann ist es notwendig, das System um zusätzliche Dokumententypen erweitern zu können, ohne dass der Quellcode der Kernkomponenten des Einweiserportals modifiziert werden muss.

R2.3: Das Einweiserportal muss um die Dokumententypen zusätzlicher Organisationseinheiten des Universitätsklinikums erweiterbar sein

Ein weiteres relevantes Pattern aus der Gruppe „Kommunikationsanforderungen“ ist das Pattern „Response Time Requirements“. Es dient der Beschreibung von Anforderungen an das Antwortzeitverhalten des zu entwickelnden Systems. Beim Einweiserportal wird ein großer Teil der Verarbeitung der Anfragen in den jeweiligen Quellsystemen der betroffenen Dokumentenarten durchgeführt. Deshalb ist das tatsächliche Antwortzeitverhalten auch vom jeweils betroffenen Quellsystem abhängig. Aus diesem Grund werden die „Response Time Requirements“ relativ zum Antwortzeitverhalten der eingebundenen Quellsysteme formuliert.

R3.1: Bei einer Suche über mehrere Quellsysteme hinweg darf die Gesamtdauer der Suche maximal 0,5 Sekunden länger als die Dauer der Suche im langsamsten der beteiligten Quellsysteme sein.

Betrachtet man die Anforderung R2.3 genauer, so lässt sich feststellen, dass diese Anforderung sowohl Aspekte der Skalierung in Form eines größeren Dokumentenanbieterkreises als auch Aspekte der Erweiterbarkeit in Form zusätzlicher Schnittstellen zu zusätzlichen Quellsystemen beinhaltet. Das nächste für die Planung des Einweiserportals relevante Pattern ist das Pattern „Extendability Requirements“. Es dient dem Zweck, Anforderungen an die inhaltliche bzw. funktionale Erweiterbarkeit des zu entwickelnden Systems zu definieren. Wie bereits in der Anforderung R2.3 definiert, ist es notwendig das Einweiserportal so um zusätzliche Dokumententypen erweiterbar zu gestalten, dass ein einfaches Wachstum um zusätzliche Organisationseinheiten des Universitätsklinikums möglich ist. Die Inhalte des Einweiserportals werden nicht in der Datenbank des Einweiserportals selbst gespeichert, sondern direkt aus den Systemen abgerufen, in denen die sie primär verwaltet werden. Daraus resultiert die Notwendigkeit, dynamisch Schnittstellen zu zusätzlichen Quellsystemen als Inhaltsanbieter zum Einweiserportal hinzufügen zu können.

R4.1: Das Einweiserportal muss um zusätzliche Quellsysteme erweiterbar sein.

Die Anforderung R1.1 legt fest, dass das Einweiserportal eine webbasierte Benutzerschnittstelle besitzen soll. Zur Verbesserung der Integration in die Arbeitsabläufe der externen Benutzer kann es hilfreich sein, einen von der verwendeten Praxissoftware unterstützten Kommunikationsstandard zu unterstützen. Da derzeit neben den, aufgrund des enthaltenen Push-All-Ansatzes kaum brauchbaren Standards PaDok D2D und VCS (vgl. Zorneck 2009) keine geeigneten Standards verfügbar sind, kann in dieser Anforderungsdefinition kein konkreter Standard für die Integration von Praxissoftware aufgenommen werden. Prinzipiell kann eine verbesserte Integration, sobald verfügbar, notwendig werden. Aus diesem Grund sind neben der gegebenen webbasierten Benutzeroberfläche Mechanismen zur Integration weiterer Schnittstellen vorzusehen.

R4.2: Das System muss um weitere Schnittstellen nach außen erweiterbar sein.

Der nächste Schritt bei der Anforderungsdefinition ist die Frage der Einfachheit der Installation der Systembestandteile. Dazu dient das Installability-Requirements-Pattern. Es besagt, dass „Installability Requirements“ mindestens aus dem beteiligten Systembestandteil, dem durchführenden Akteur und einer Beschreibung der erforderlichen Einfachheit der Systeminstallation bzw. -aktualisierung bestehen sollen (vgl. Withall 2007:S. 275). Das Einweiserportal setzt sich in dieser Betrachtungsweise primär aus einem Clientbestandteil, genannt Benutzerschnittstelle, und einem Serverbestandteil zusammen. In Kapitel 6.3.1 wurde ermittelt, dass bei den externen Benutzern in Kreiskrankenhäusern und Praxen niedergelassener Ärzte nur geringe administratorische und IT-technische Kenntnisse vorausgesetzt werden können. Insofern erscheint es erforderlich, dass die Installation der Clientbestandteile einfach genug ist, dass sie auch mit diesen geringen Kenntnissen problemlos durchgeführt werden kann.

R5.1: Die Benutzerschnittstelle für die Endbenutzer soll so gestaltet sein, dass für ihre Nutzung keinerlei zusätzliche Softwareinstallation auf dem Clientsystem notwendig ist.

Anders sieht die Situation bei den Serverbestandteilen des Einweiserportals aus. Sie werden am Universitätsklinikum betrieben, wo eine umfangreiche Mannschaft aus Administratoren und Entwicklern verfügbar ist. Deshalb ist eine präzise und eindeutige Installationsdokumentation für eine problemlose Durchführung der Installation und Aktualisierung der Serverbestandteile ausreichend.

R5.2: Die Serverbestandteile am Universitätsklinikum Regensburg benötigen eine geeignete Installationsdokumentation.

Das letzte der Patterns für die Definition der Anforderungen an die Flexibilität des Einweiserportals ist das Pattern „Unparochialness Requirements“ (Unbeschränktheitsanforderungen), das sich mit den Anforderungen zu den Eigenschaften beschäftigt, bezüglich derer das System nicht auf eine bestimmte Umgebung beschränkt sein darf (vgl. Withall 2007:S. 254, Applicability). Für das Einweiserportal gilt allgemein, dass sein Einsatz nicht auf die aktuelle Systemumgebung des Universitätsklinikums Regensburg beschränkt sein darf. Es muss sowohl in einer veränderten Umgebung innerhalb des Universitätsklinikums als auch in der Umgebung eines beliebigen anderen deutschen Krankenhauses einsetzbar sein. Das bedeutet,

dass verschiedene Systemkomponenten unterschiedlicher Umgebungen unabhängig vom Einweiserportal eingesetzt und im Einweiserportal verwendet werden können müssen. Was wiederum dazu führt, dass nicht nur die Schnittstellen zu medizinischen Primärsystemen (S3 bis S5), sondern auch die Schnittstellen zu Authentifizierungskomponenten, zur Benutzerverwaltung oder zum zentralen Logging (S6 und S7) beliebig austauschbar sein müssen. Beim Einweiserportal gilt das insbesondere für die Benutzerverwaltung, da zum Zeitpunkt der Entwicklung des Portals am Universitätsklinikum das Produkt eDirectory von Novell eingesetzt wurde, dessen Marktanteile stetig sinken. Somit ist hier mittelfristig ein Austausch durch ein verbreiteteres Produkt zu erwarten.

R6.1: Das verwendete Benutzerverwaltungssystem muss austauschbar sein.
R6.2: Die verwendeten Speicherungsorte für Anwendungslogs müssen austauschbar sein.

### 6.3.3. Spezielle Rahmenbedingungen für die Sicherheit des Einweiserportals

Die Definition der Sicherheitsanforderungen hängt neben den in Kapitel 6.3.1 erarbeiteten Rahmenbedingungen von weiteren Eigenschaften der Umgebung des zu entwickelnden Systems ab. Deshalb ist vor der Ermittlung konkreter Sicherheitsanforderungen eine zusätzliche Analyse dieser Rahmenbedingungen nötig. Die hierfür vorgesehenen Patterns können in der Ebene *Anforderungsanylse* des Schwerpunkts *Sicherheit* ermittelt werden. Der Ausgangspunkt dazu ist das Pattern „Security Needs Identification for Enterprise Assets“. Die Ermittlung konkreter Anforderungen an die Sicherheit des Einweiserportals erfolgt in Kapitel 6.3.4 durch die Patterns der Gruppe „Ermittlung der Sicherheitsanforderungen“. Die Menge der relevanten Patterns besteht aus allen, durch ihre Beziehungen zwischen das Pattern „Security Needs Identification for Enterprise Assets“ und die Gruppe „Ermittlung der Sicherheitsanforderungen“ eingeordneten Patterns. Dabei handelt es sich um die Patterns:

- Security Needs Identification for Enterprise Assets
- Analyse der Organisationsstruktur
- Asset Valuation
- Threat Assessment
- Risk Determination
- Enterprise Security Approaches
- Enterprise Partner Communication
- Enterprise Security Services.

Das Ausgangspattern ist, wie bereits geschildert, das Pattern „Security Needs Identification for Enterprise Assets“. Es umfasst die in vier Schritte gegliederte Analyse der sicherheitsrelevanten Bestandteile des zu untersuchenden Geschäftsbetriebs, einschließlich verschiedener Einflussfaktoren auf den Sicherheitsbedarf und die Arten des notwendigen Schutzes. Bezogen auf das Einweiserportal lassen sich am Universitätsklinikum Regensburg die Patientendaten, das Kliniknetz und primäre IT-Systeme wie das Krankenhausinformationssystem (KIS), Bildarchive (PACS) oder Dokumentenarchive identifizieren. Ihr Schutzbedarf wird durch verschiedene Faktoren wie z. B. die gesetzlichen Rahmenbedingungen zum medizinischen

Datenschutz, das Vertraulichkeitsbedürfnis der Patienten, die erforderlich reibungslose Patientenversorgung ohne Systemausfälle, aber auch das für das Vertrauen der Patienten notwendige öffentliche Ansehen beeinflusst. Die genannten Einflussfaktoren auf den Schutzbedarf betreffen die ebenfalls schon genannten materiellen und immateriellen Bestandteile der Versorgungseinrichtung unterschiedlich stark. So betrifft das Datenschutzrecht insbesondere den Schutzbedarf der Patientendaten. Das öffentliche Ansehen ist sowohl von der Sicherheit der Patientendaten als auch der primären IT-Systeme und des Kliniknetzes abhängig. Das Vertraulichkeitsbedürfnis der Patienten fokussiert in erster Linie die Patientendaten, während die reibungslose Patientenversorgung ohne Systemausfälle sichere primäre IT-Systeme und ein gut abgesichertes Kliniknetz voraussetzt. Aus diesen Rahmenbedingungen resultiert die folgende Liste allgemeiner Sicherheitsanforderungen:

R7.1: Sowohl lesender als auch schreibender Zugriff auf Patientendaten darf nur berechtigten Personen gestattet sein.
R7.2: Bei einer widerrechtlichen Verbreitung von Patientendaten durch zum Zugriff berechtigte Nutzer muss der verbreitende Nutzer nachweisbar sein.
R7.3: Ein direkter Zugriff auf das Kliniknetz darf nicht möglich sein.
R7.4: Der indirekte Zugriff auf das Kliniknetz darf nur berechtigten Personen möglich sein.
R7.5: Eine Manipulation von Daten in den primären IT-Systemen darf von außerhalb des Kliniknetzes nicht möglich sein.
- Sowohl nicht durch Personen ohne Nutzerkonto
- als auch nicht durch berechtigte Nutzer des Einweiserportals.

R7.6: Jede Nutzung des Einweiserportals muss nachvollziehbar sein.

Das Pattern „Security Needs Identification for Enterprise Assets“ hat ausgehende Beziehungen zu den Patterns „Asset Valutation“, „Vulnerability Assessment“ und „Thread Assessment“. Darüber hinaus besitzt es über die Beziehung zur Gruppe „Analyse der Sicherheitssituation“ zusätzlich eine Beziehung zum Pattern „Analyse der Organisationsstruktur“. Die Reihenfolge der Betrachtung der vier genannten Patterns ist frei wählbar, da zwischen diesen Patterns keine Abhängigkeiten bestehen. Aus diesem Grund folgt als Nächstes die Analyse der Organisationsstruktur. Sie hängt zusätzlich von den Ergebnissen der Analyse der betroffenen Behandlungspfade und den Ergebnissen der Analyse der organisatorischen Verteilung im Verbund ab. Da in der bisherigen Analyse die Definition konkreter einrichtungsübergreifender Behandlungspfade ausgeschlossen wurde sind nur die Schritte 1 und 2 des Patterns „Analyse der Organisationsstruktur“ sinnvoll anwendbar.

Bezogen auf das Einweiserportal lassen sich die folgenden Akteure identifizieren:

Akteur 1: Niedergelassener Arzt
Akteur 2: Arzt in externem Krankenhaus
Akteur 3: Arzt am Universitätsklinikum.

Die Anwendung des Patterns „Asset Valutation“ ist nicht erforderlich, da alle identifizierten Risiken unter allen Umständen auszuschließen sind, weshalb eine weitere Gewichtung keinen zusätzlichen Erkenntnisgewinn für die Entwicklung der Software bietet.

Der nächste Schritt ist die Anwendung des Patterns „Threat Assessment“. Es umfasst die Identifikation und Analyse von möglichen Bedrohungen und der Quellen dieser Bedrohungen. Für das Einweiserportal können die Bedrohungen in die Gruppen allgemeine Bedrohungen des Internets und potenzielle Bedrohungen, die speziell durch das Einweiserportal selbst entstehen können, aufgeteilt werden. Aus einer Schnittstelle zum Internet resultieren Bedrohungen wie gezielte oder massenhaft ausgeführte Hackerangriffe, das Einführen von Malware (z. B. Viren) in das Kliniknetz oder das Mitlesen von Daten bei der Übertragung. Ihre Auswirkungen reichen vom unberechtigten Lesen von Patientendaten bis hin zur Manipulation von Inhalten in den primären IT-Systemen und zur Störung des Behandlungsbetriebs. Die Bedrohungen, die direkt aus dem Betrieb des Einweiserportals resultieren sind hauptsächlich Bedrohungen, die durch authentifizierte Benutzer des Einweiserportals entstehen. Sie schließen das Risiko der unberechtigten Benutzung von Benutzerkonten ein. Sie umfassen die Manipulation von Patientendaten in den primären IT-Systemen durch Portalbenutzer von außerhalb des Kliniknetzes, die Weitergabe von Patientendaten durch Portalbenutzer und das unberechtigte Lesen oder Schreiben von Daten. Dabei kann die Manipulation von Patientendaten zu Störungen in den Betriebsabläufen und Behandlungsfehlern durch Fehlinformation führen. Die Weitergabe von Patientendaten hat den Verlust der Vertraulichkeit und, bei bekannt werden juristische Konsequenzen, sowie einen Verlust von Ansehen für das Universitätsklinikum zur Folge.

Auf die Analyse möglicher Bedrohungen folgt ihre Bewertung mittels des Patterns „Risk Determination“. Es baut direkt auf den per „Threat Assessment“ ermittelten Bedrohungen auf und kombiniert sie mit einer Risikobewertung und möglichen Ursachen. Die Risikobewertung führt zu einer Klassifikation der Bedrohungen nach hohem, mittlerem oder niedrigem Risiko. Bei der Klassifikation sind sowohl die Wahrscheinlichkeit des Auftretens als auch der Umfang der zu erwartenden Folgen zu berücksichtigen. Um Übersichtlichkeit und eine gute Lesbarkeit zu gewährleisten wird das Ergebnis in einer in zwei Ebenen gegliederten Auflistung dargestellt. In der ersten Ebene werden die Bedrohungen einschließlich einer am Ende in Klammern beigefügten Risikobewertung aufgelistet. In der zweiten Ebene darunter werden mögliche Ursachen für das Auftreten dieser Bedrohungen angegeben.

**Risiko 1:** Unberechtigtes Lesen von Patientendaten durch reguläre Portal-Benutzer (hoch)
- ***Risiko 1-1:*** Durch unzureichendes Berechtigungskonzept
- ***Risiko 1-2:*** Durch fehlende Berechtigungsaudits
- ***Risiko 1-3:*** Durch unzureichende technische Berechtigungsprüfung

**Risiko 2:** Eindringen in das Kliniknetz durch Dritte aus dem Internet (hoch)
- ***Risiko 2-1:*** Durch Mängel im Abschottungskonzept
- ***Risiko 2-2:*** Durch Sicherheitslücken in der Software der Sicherheitskomponenten
- ***Risiko 2-3:*** Durch schwaches Authentifizierungsverfahren

**Risiko 3:** Mitlesen übertragener vertraulicher Daten (z. B. Patientendaten) (hoch)
- ***Risiko 3-1:*** Durch fehlende Verschlüsselung

**Risiko 4:** Unberechtigtes Verwenden eines schlecht gesicherten Benutzerkontos durch Dritte aus dem Internet (hoch)
- ***Risiko 4-1:*** Durch schwaches Authentifizierungsverfahren (z. B. schwaches Passwort)
- ***Risiko 4-2:*** Durch Diebstahl eines Hardware-Authentifizierungstokens

**Risiko 5:** Unberechtigtes oder unbeabsichtigtes Verändern oder Manipulieren von Patientendaten in den primären IT-Systemen durch reguläre Portalbenutzer (mittel)

***Risiko 5-1:*** Durch unzureichende Berechtigungsprüfung

***Risiko 5-2:*** Durch Software- und Bedienungsfehler

***Risiko 5-3:*** Durch Ausnutzen von Sicherheitslücken in der Portalimplementierung

**Risiko 6:** Unberechtigte Weitergabe von Patientendaten an Dritte durch reguläre Portal-Benutzer (mittel)

***Risiko 6-1:*** Mittels der Funktion Kopieren und Einfügen

***Risiko 6-2:*** Durch Weitergabe gespeicherter Dateien

***Risiko 6-3:*** Durch Weitergabe des System-Accounts durch Benutzer.

Das Pattern „Risk Determination“ besitzt ausgehende Beziehungen zu den Patterns „Enterprise Security Approaches“ und „Enterprise Partner Communication“ sowie zu der Gruppe „Ermittlung der Sicherheitsanforderungen“. Letztere besitzt zudem eingehende Beziehungen aus den Patterns „Enterprise Partner Communication“ und „Enterprise Security Services“. „Enterprise Security Services“ hängen wiederum von den Ergebnissen des Patterns „Enterprise Security Approaches“ ab. Daraus resultiert als weitere Vorgehensweise die Anwendung des Patterns „Enterprise Security Approaches“ gefolgt von „Enterprise Security Services“ und „Enterprise Partner Communication“.

Das Pattern „Enterprise Security Approaches“ greift direkt die im Rahmen der „Risk Determination“ ermittelten Risiken auf und kombiniert diese mit geeigneten Lösungsansätzen. Die Lösungsansätze lassen sich grob in die Gruppen *Vorsorge*, *Erkennung* und *Reaktion* zusammenfassen (vgl. Schumacher u. a. 2005:S. 161). Die für die Risiken des Einweiserportals identifizierten Lösungsansätze lassen sich in die Gruppe *Vorsorge* einordnen. Die Zuordnung der Lösungen erfolgt auf Basis der für die Darstellung der „Risk Determination“ gewählten Nummerierung.

**Risiko 1-L:** Unterbinden unberechtigter Lesezugriffe durch feingranulares Berechtigungskonzept

**Risiko 2-L:** Mindern der Wahrscheinlichkeit eines Eindringens in das Kliniknetz durch eine Netzwerkarchitektur mit DMZ und einem Single Point of Access

**Risiko 3-L:** Erhaltung der Vertraulichkeit übertragener Daten durch verschlüsselte Kommunikation zwischen Einweiserportal und Benutzer

**Risiko 4-L:** Verhindern einer unberechtigten Verwendung von Benutzerkonten durch mehrfaktorige oder anderweitig starke Authentifizierungsverfahren

**Risiko 5-L:** Verhindern von Veränderungen oder Manipulationen von Patientendaten durch ausschließliche Implementierung lesender Operationen im Einweiserportal

**Risiko 6-L:** Eine unberechtigte Weitergabe von Patientendaten an Dritte durch reguläre Portal-Benutzer kann nicht vollständig unterbunden werden.

Aus den aufgelisteten „Enterprise Security Approaches“ resultieren konkrete „Enterprise Security Services“. Sie sind Dienste, die zur Gewährleistung adäquater Sicherheit z. B. als wiederverwendbare Komponenten für die Konzeption des restlichen Systems zur Verfügung stehen.

Für das Einweiserportal sind die folgenden vier Dienste als relevant identifizierbar:

- Authentifizierungsdienst
- Autorisierungsdienst
- Benutzerverwaltung
- Dienst zur Verwaltung von Logausgaben.

Das Pattern „Enterprise Partner Communication“ dient der Beschreibung der Kommunikation mit Partnerunternehmen oder Partnereinrichtungen, und zwar so, wie sie bei einem bestehenden System bereits abläuft oder bei einem noch zu entwickelnden System zukünftig ablaufen soll. Da sich das Einweiserportal zum Zeitpunkt der Anwendung des Patterns erst in der Phase der Planung befindet, ist der Sollzustand zu beschreiben. Der Zweck des gesamten Systems *Einweiserportal* ist die Kommunikation von Patientendaten mit Partnereinrichtungen. Aus diesem Grund ist das gesamte Einweiserportal von der Kommunikation mit anderen Einrichtungen geprägt. Die Partner sind, wie bereits in Kapitel 6.3.1 beschrieben, andere Krankenhäuser und niedergelassene Ärzte. Die Kommunikation mit diesen Partnern erfolgt über das unsichere Netz Internet. Diese Verbindung ist durch eine Lösung zur verschlüsselten Kommunikation (z. B. ein Virtual Private Network (VPN)) zu sichern. Der Zugriff hat über einen zentralen Zugriffspunkt, die Kombination aus einer VPN-Gateway und dem Server des Einweiserportals zu erfolgen. Die Authentifizierung von Benutzern erfordert zwei unterschiedliche Faktoren, den Besitz eines Hardwaretokens und das Wissen eines Passworts. Alle authentifizierten Kommunikationspartner besitzen nur lesenden Zugriff auf die für Sie bestimmten Daten. Die zur Verfügung stehenden Inhalte sind vielfältig und können über Methoden zur freien Suche oder über eine Liste aktueller Dokumente gefunden und über eine Methode für den lesenden Zugriff abgerufen und angezeigt werden. Zur Anzeige erfolgt eine Übertragung des Inhalts auf den Rechner des Kommunikationspartners. Die Berechtigung zum Zugriff auf die Daten im Einweiserportal kann für jeden Benutzer einzeln und zu jeder Zeit direkt widerrufen werden.

### 6.3.4. Anforderungen an Sicherheit und Zuverlässigkeit

Neben den Anforderungen bezüglich der Kommunikation und der Flexibilität sind die Anforderungen bezüglich der Sicherheit und der Zuverlässigkeit der zweite Teilbereich der Anforderungsdefinition für das Einweiserportal des Universitätsklinikums Regensburg. Sie basieren sowohl auf den in Kapitel 6.3.1 definierten Rahmenbedingungen von Kommunikation und Flexibilität als auch auf den speziellen Rahmenbedingungen für die Sicherheit des Einweiserportals (siehe Kapitel 6.3.3). Die zur Definition von Anforderungen bezüglich der Sicherheit und Zuverlässigkeit des Einweiserportals notwendigen Patterns lassen sich durch Zusammenfassen der beiden Gruppen „Ermittlung der Sicherheitsanforderungen“ und „Ermittlung der Zuverlässigkeitsanforderungen“, reduziert um erkennbar nicht relevante oder bereits beantwortete Fragestellungen ermitteln.

Patterns zur Anforderungsdefinition bezüglich Sicherheit und Zuverlässigkeit:

- Access Control Requirements (Anhang S. 248)
- Audit Requirements (Anhang S. 282)

- Audit Trails and Logging Requirements (Anhang S. 284)
- I&A Requirements (Anhang S. 400)
- Intrusion Detection Requirements (Anhang S. 415)
- Non-Repudiation Requirements (Anhang S. 457)
- Dynamic Capacity Requirements (Anhang S. 353)
- Throughput Requirements (Anhang S. 589).

Das erste der anzuwendenden Patterns ist das Muster „Access Control Requirements“, da bei der Anwendung des Patterns „Enterprise Security Services“ die Bereitstellung eines Autorisierungsdienstes als notwendig identifiziert worden ist. Das Access-Control-Requirements-Pattern dient der Spezifikation konkreter Zugriffskontrollanforderungen, die unter anderem Anforderungen an den zu gestaltenden Autorisierungsdienst sind. Als Ausgangsinformation für die Definition der Anforderungen dienen die Ergebnisse der Patterns „Analyse der Organisationsstruktur“, „Risk Determination“ und „Enterprise Partner Communication“. Das Ergebnis der Anwendung des Patterns „Access Control Requirements“ ist eine Auflistung von konkreten Anforderungen an Zugriffsbeschränkungen, die das Einweiserportal besitzen muss. Unter Zugriffsbeschränkungen sind sowohl physikalische Einschränkungen des Zugriffs über einen oder mehrere definierte netzwerkseitige Zugriffspunkte als auch inhaltliche Zugriffseinschränkungen zu verstehen. Bezogen auf das Einweiserportal ergibt sich die folgende Auflistung von „Access Control Requirements“:

R8.1: Nur am zentralen Zugriffspunkt (siehe Risiko 2-L) authentifizierte Benutzer erhalten Zugriff auf das System.
R8.2: Das Einweiserportal bietet auf alle enthaltenen Dokumente nur lesenden Zugriff.
R8.3: Die individuellen Berechtigungen werden pro Dokumentenart mittels eigener Mechanismen durchgesetzt.
R8.4: Nur Benutzer, die ein Dokument oder abhängiges Dokument auch in Papier erhalten würden, erhalten automatisch lesenden Zugriff auf dieses Dokument im Einweiserportal.
R8.5: Die Verwaltung der Zugriffsberechtigungen erfolgt im Erstellungsprozess des Dokuments durch seinen Ersteller.
R8.6: Ein späteres Hinzufügen von Zugriffsberechtigungen auf ein Dokument darf nur durch berechtigte Benutzer möglich sein.

Ein weiterer, als erforderlich identifizierter Enterprise Security Service umfasst das Logging von Systemaktivitäten. Systemlogs können sowohl zur Analyse von Fehlern in der Software als auch zur Analyse von Sicherheitsvorfällen eingesetzt werden. Notwendige Fähigkeiten, die ein System bezüglich der Analysierbarkeit von Sicherheitsvorfällen besitzen soll, können mit Hilfe des Audit-Requirements-Patterns spezifiziert werden. Desweiteren basiert die Definition der „Audit Requirements“ noch auf Ergebnissen aus der „Analyse der Organisationsstruktur“, der „Risk Determination“ und der Beschreibung der „Enterprise Partner Communication“. Für das Einweiserportal lassen sich die folgenden fünf Anforderungen identifizieren, die unter anderem auf Anforderungen aus dem medizinischen Datenschutz zurückzuführen sind:

R9.1: Es muss möglich sein, die Dokumentenzugriffe von Benutzern auszuwerten.
R9.2: Es muss möglich sein, die Veränderungen in Benutzerberechtigungen über den Zeitverlauf sowohl nach dem betroffenen Benutzer als auch dem Änderer auszuwerten.
R9.3: Auswertungen über im System aufgetretene Ausnahmen und den zugehörigen Benutzerkontext müssen möglich sein.
R9.4: Zeitpunktbezogene Auswertungen über am System angemeldete Benutzer müssen möglich sein.
R9.5: Alle Basisdaten für die Anforderungen R9.1 bis R9.4 müssen zuverlässig und belastbar sein.

Die Audit Requirements R9.1 bis R9.4 bestimmen die Aspekte der Auditierbarkeit des Einweiserportals. Um diesen Zustand zu erreichen, müssen während des Systembetriebs fortlaufend Daten über verschiedene Systemaktivitäten erhoben werden. Dieses Vorgehen wird als Logging bezeichnet. Das Pattern „Audit Trails and Logging Requirements“ dient sowohl der Definition von Wegen, über die entsprechende Audit-Maßnahmen durchgeführt werden können, als auch der Definition von Ereignissen, die im Log des Systems festgehalten werden müssen. Für das Einweiserportal sind ausschließlich Anforderungen bezüglich Ereignissen, die im System-Log des Einweiserportals festzuhalten sind, erfasst. Sie besitzen direkte Beziehungen zu den Anforderungen R9.1 bis R9.5 und sind selbst mit den Nummern R10.1 bis R10.5 gekennzeichnet.

R10.1: Alle An- und Abmeldungen von Benutzern müssen mit Benutzerkennung und Zeitpunkt gespeichert werden. (Bezieht sich auf R9.4)
R10.2: Alle Zugriffe von Benutzern auf Dokumente müssen mit Zeitpunkt, Benutzer und Dokumenten-ID sowie ggf. eine Dokumentenversion gespeichert werden. (R9.1)
R10.3: Alle Veränderungen an Benutzern, Benutzerrollen und Benutzermappings werden mit Benutzer, Zeitpunkt und Änderer erfasst. (R9.2)
R10.4: Alle auftretenden Systemausnahmen müssen mit Fehlermeldung, vollständigem Stacktrace, Benutzer und dem Zeitpunkt des Fehlerauftretens erfasst werden. (R9.3)
R10.5: Die Logging-Komponente bietet nur je eine Methode zum Schreiben neuer Logeinträge und zum Lesen bestehender Einträge. Die Methode zum Schreiben neuer Einträge ist nur durch das Einweiserportal selbst aufrufbar und erfordert die Berechtigung eines angemeldeten Benutzers. Das Lesen von Einträgen ist nur mit administratorischen Berechtigungen erlaubt. (R9.5)

Die Prüfung von Zugriffsrechten, wie sie zu Beginn dieses Kapitels mit dem Access-Control-Requirements-Pattern definiert werden, bedingt die zuverlässige Identifikation und Authentifizierung des zugreifenden Benutzers. Aus diesem Grund existiert eine von dem Pattern „Access Control Requirements“ zu dem Pattern „I&A Requirements“ führende Beziehung. Sie weist bei der Definition der Access Control Requirements bereits auf die Notwendigkeit der Definition von Anforderungen bezüglich der Benutzerauthentifizierung hin (siehe Abhängigkeitsgraph auf S. 400). Außerdem beeinflusst die Qualität des gewählten Authentifizierungsmechanismus auch die Belastbarkeit des Benutzerbezugs der für mögliche Audits festgehaltenen Systemdaten. Über die Gruppe „Ermittlung der Sicherheitsanforderungen“ wird die Definition der „I&A Requirements“ primär durch die Patterns „Analyse der Organi-

sationsstruktur“, „Risk Determination“, „Enterprise Partner Communication“ und „Enterprise Security Services“ beeinflusst. Da das Einweiserportal patientenbezogene Daten über das Internet für berechtigte Benutzer zur Verfügung stellen soll (vgl. Enterprise Partner Communication) kommt ein einfacher, nur auf einem Kriterium basierender Authentifizierungsmechanismus nicht in Frage.

R11.1: Benutzer dürfen sich nur nach einer starken, von mehreren Faktoren abhängigen Authentifizierung über das Internet am Einweiserportal anmelden können.

Da von den Berechtigungen, die Administratoren am Einweiserportal besitzen, eine besondere Gefährdung ausgehen kann ist ihre Anmeldung zusätzlich einzuschränken.

R11.2: Administratoren dürfen sich nur von wenigen ausgewählten Arbeitsstationen aus dem Kliniknetz an der Administrationsoberfläche des Einweiserportals anmelden. Eine Anmeldung über das Internet ist auszuschließen.

Um das Einweiserportal aus dem Internet erreichbar zu machen, muss eine, wenn auch sicherheitstechnisch stark eingeschränkte, direkte oder indirekte Verbindung zwischen Internet und Einweiserportal und somit zwischen Internet und Kliniknetz aufgebaut werden. Da das Fehlen einer Verbindung zwischen Internet und Kliniknetz den größtmöglichen Schutz vor unberechtigten Zugriffen aus dem Internet bietet, ist jede Form der Verbindung eine Verschlechterung dieses Zustands. Eine Kommunikation von Befunden mit externen Einrichtungen ist ohne Öffnen des Netzes nicht möglich. Deshalb muss zusätzlich zu den Maßnahmen der Abschottung und Zugriffsrestriktion eine Überwachung stattfinden, mit der sich eventuelle Umgehungen der Sicherheitsmaßnahmen erkennen lassen. Anforderungen an die Erkennung solcher Einbrüche können mit Hilfe des Intrusion-Detection-Requirements-Patterns ermittelt und definiert werden. Für das Einweiserportal gilt die folgende Anforderung.

R12.1: Die VPN-Gateway, die das Einweiserportal mit dem Internet verbindet, muss unautorisierte Zugriffsversuche mit analysierbaren Daten festhalten, sodass diese im Rahmen eines erweiterten Sicherheitsaudits ausgewertet werden können.

Ein weiteres Pattern zur Definition von Anforderungen, die für das Einweiserportal relevant sind, ist das Non-Repudiation-Requirements-Pattern. Es beschreibt die Definition von Anforderungen an die Qualität von Aufzeichnungen und Nachweismechanismen zur Aufklärung von Sicherheitsvorfällen. Die zugehörigen Anforderungen beschreiben, bei welchen Aufzeichnungen oder Nachweismechanismen es besonders schwer sein muss, ihre Aussagen zu widerlegen oder in Zweifel zu ziehen. Für das Einweiserportal gelten zu diesem Aspekt die folgenden drei Anforderungen.

R13.1: Logeinträge über Systemzugriffe dürfen nicht als, durch Benutzer manipulierbar in Frage gestellt werden können.

R13.2: Bei Entdecken eines illegal weitergegebenen medizinischen Dokuments muss der Weitergebende zweifelsfrei identifizierbar sein. Wenn die Information zum Ersteller des Dokuments manipuliert wurde, muss das zweifelsfrei erkennbar sein.

R13.3: Es muss zweifelsfrei ermittelt werden können, ob es sich bei einem Dokument um das vom Portal ausgelieferte Original oder um eine manipulierte Kopie handelt.

Die Zuverlässigkeit eines Systems hängt neben den verschiedenen zugleich sicherheitsrelevanten Faktoren auch von seiner Verfügbarkeit und Belastbarkeit ab. Die Belastbarkeit des Systems ist von der Leistungsfähigkeit, die das System bezüglich verschiedener Faktoren bietet, abhängig. Die statische Kapazität ist die Fähigkeit, eine bestimmte Menge von Inhalten dauerhaft speichern zu können. Sie ist für die Verfügbarkeit des Einweiserportals von untergeordneter Bedeutung, da seine Aufgabe nicht darin besteht, Daten zu speichern, sondern Daten aus dritten Systemen zu vermitteln und zu übertragen. Aus diesem Grund ist das Pattern „Static Capacity Requirements“ (siehe Anhang S. 581) nicht Bestandteil der weiteren Anforderungsanalyse.

Im Gegensatz zur statischen Kapazität ist die dynamische Kapazität für die Zuverlässigkeit des Einweiserportals ein relevanter Faktor. Mit dem Pattern „Dynamic Capacity Requirements“ können Anforderungen bezüglich der Menge parallel möglicher Benutzersitzungen definiert werden. Für das Einweiserportal gilt, dass mindestens 30 parallele Benutzersitzungen unter Einhaltung der definierten „Response Time Requirements“ bedient werden können müssen (R14.1). Ähnlich zu den „Dynamic Capacity Requirements“ sind die „Throughput Requirements“. Sie dienen der Spezifikation des erforderlichen Durchsatzes bestimmter Anfragetypen. Das kann sowohl eine Menge von Anfrage bedeuten, die innerhalb einer definierten Zeit vom System zu bearbeiten sind, als auch eine Frequenz oder Zeit in der das System Ergebnisse liefern muss (vgl. Withall 2007:S. 204). Da das Einweiserportal selbst nicht für den vollständigen Prozess der Ermittlung und den Zugriff auf Dokumente zuständig ist, ist diese Anforderung schwer definierbar. Antwortzeitverhalten und der mögliche Durchsatz der Backend-Systeme, aus denen die Dokumente geladen werden, variieren stark und können so das Ergebnis in nicht vorhersehbarer Form beeinflussen. Aus diesem Grund ist für das Einweiserportal eine Beschreibungsform gewählt worden, die eine von den Backend-Systemen unabhängige Beschreibung der „Throughput Requirements“ zulässt.

R15.1: Für jedes Inter System Interface (S3-S5) gilt unter Berücksichtigung der „Dynamic Capacity Requirements“ (vgl. R14.1), dass die Verarbeitung von Suchanfragen und die Dokumentenübermittlung im Portal zusätzlich zur Verarbeitungsdauer in den angebundenen Backend-Systemen maximal 0,5s betragen darf.

R15.1 korrespondiert mit dem Response Time Requirement R3.1 und ergänzt es um den Bezug zu einer Kapazität, bei der das Verhalten zu erreichen ist. Dennoch liefert R15.1 aufgrund der unklaren Umgebungssituation mit unterschiedlich schnellen Quellsystemen kein klassisches „Throughput Requirement“.

Die Sammlung der mit den verschiedenen Patterns der Ebene *Anforderungsanalyse* ermittelten Anforderungen beschreibt beschreibt bezüglich einer Vielzahl von Aspekten notwendige

Eigenschaften des Einweiserportal. Die systematische Auswahl der Patterns stellt sicher, dass diese Aspekte in Abhängigkeit des Anwendungsgebiets der zu entwickelnden Software bewertet und gegebenenfalls berücksichtigt werden. Das Ergebnis ist eine dem Problem angemessene Sammlung von Anforderungen, die in die Gestaltung der Softwarearchitektur und des Softwaredesigns eingehen können.

## 6.4. Softwarearchitektur und Softwaredesign

Im Entwicklungsprozess des Einweiserportals folgt auf die Anforderungsanalyse und Anforderungsdefinition die Auswahl der Patterns zur Gestaltung der Softwarearchitektur und des Softwaredesigns. Dazu werden, wie in Kapitel 6.1 beschrieben, ausgehend von den bereits angewendeten Patterns aus der Ebene *Anforderungsanalyse* abhängige Patterns der darunterliegenden Ebenen betrachtet und abhängig von ihrer Eignung in die Softwarearchitektur oder das Softwaredesign einbezogen.

Ausgehend von dem Einstiegspattern der Anforderungsanalyse, „Analyse der organisatorischen Verteilung im Verbund“, ist eine Beziehung zur Gruppe „Verteilungsarchitektur der Akte“ definiert. Sie visualisiert, dass die Beschreibung der organisatorischen Rahmenbedingungen eine Voraussetzung für die Auswahl einer geeigneten Verteilungsarchitektur für das Einweiserportal ist. Weitere Entscheidungsgrundlagen liefern die Patterns der Gruppe „Kommunikationsanforderungen“, insbesondere das Pattern „Inter-System Interface Requirements“. Für die Gestaltung der Verteilungsarchitektur stehen in der Gruppe die Patterns „Akte mit zentraler Datenspeicherung“ (siehe Anhang S. 264), „Akte mit regional zentralisierter Datenspeicherung“ (siehe Anhang S. 257) und „Akte mit vollständig dezentraler Datenspeicherung“ (siehe Anhang S. 260) zur Auswahl. Aufgrund der Ergebnisse der Anforderungsanalyse in den Patterns „Analyse der organisatorischen Verteilung im Verbund“ und „Inter-System Interface Requirements“ besteht eine der zentralen Anforderungen darin, dass die Speicherung der Daten in den bisherigen Primärsystemen unverändert weitergeführt wird. Diese Anforderung entspricht den Aussagen in den Abschnitten *Kontext* und *Problem* des Patterns „Akte mit vollständig dezentraler Datenspeicherung“. Aus diesem Grund ist für die Gestaltung der Verteilungsarchitektur das Pattern „Akte mit vollständig dezentraler Datenspeicherung“ auszuwählen. Das Pattern selbst beschreibt zwei potenzielle Lösungswege zur Gestaltung einer Akte mit dezentraler Datenspeicherung. Während Ansatz 1 die Verwendung von asynchroner Kommunikation und Synchronisationsmechanismen beschreibt, beinhaltet Ansatz 2 die Umsetzung mit Mitteln der synchronen Kommunikation unter Verwendung des Pull-Prinzips. Angesichts der bei der Anwendung des Inter-System-Interface-Requirements-Patterns beschriebenen Schnittstellen ist Ansatz 2 zu wählen, da alle in den bestehenden Systemen bereits verfügbaren Schnittstellen aufgrund ihrer Basistechnologien wie SAP RFC oder XML-RPC synchrone Kommunikation erfordern.

Die Auswahl des Patterns „Akte mit vollständig dezentraler Datenspeicherung“ macht im nächsten Schritt die Definition der Verteilung der Dokumentenablage pro Patient notwendig. Nur für Akten mit zentraler Datenspeicherung ist keine Festlegung der Verteilung der Dokumentenablage pro Patient nötig, da alle Akten unabhängig vom Patienten nicht verteilt auf einem zentralen System vorliegen. Die Betrachtung der Ergebnisse der Untersuchung betroffener Subsysteme unter Berücksichtigung bestehender Unparochialness und Scalability

Requirements ergibt, dass die bestehenden Systeme is.h.med und Easy sowie das PACS jeweils Teile der elektronischen Akte eines Patienten enthalten. Keines der Systeme beinhaltet institutionsbezogen vollständige Akten, die unabhängig von den anderen Systemen sind. Deshalb ist für das Einweiserportal nur eine pro Patient verteilte Dokumentenablage möglich. Diese wird durch das Pattern „Fully Distributed Patient Documents“ (siehe Anhang S. 394) beschrieben.

Anforderung R4.2 ist eine aus der Anwendung des Extendability-Requirements-Patterns resultierende Anforderung. Sie beschreibt dass es erforderlich ist, das System um weitere, nach außen gerichtete Schnittstellen erweitern zu können. Das kann z. B. zur Erfüllung geforderter Schnittstellen oder Standards für die Kommunikation mit der Praxissoftware der niedergelassenen Ärzte oder den Krankenhausinformationssystemen der Kreiskrankenhäuser nötig sein. Um dieser Anforderung gerecht zu werden, ist in die Softwarearchitektur des Einweiserportals zwischen der Portaloberfläche und der Logik zur Verarbeitung der Anfragen das Multi-Channel-Access-Provider-Pattern (siehe Anhang S. 452) einzubringen. Es beschreibt einen Ansatz, zentrale Funktionalität durch unterschiedliche Adapter für verschiedene Arten von Schnittstellen und Standards verfügbar zu machen. Der Lösungsansatz „Multi-Channel Access Provider“ besteht aus zwei Schichten (Schichtenarchitektur Muster), von denen in der unteren Schicht (Basisschicht) generisch die eigentliche Funktionalität implementiert wird, während die darüberliegende Schicht (Schnittstellenschicht) aus einer Vielzahl von Fassade- bzw. Adapter-Implementierungen besteht, die unter Verwendung der Methoden der Basisschicht die Verhaltensweisen der verschiedenen adressierten Standards adaptieren.

Das System-Kontext-Diagramm der „Inter System Interface Requirements“ stellt im Bereich der Schnittstellen S3 bis S5 eine Menge ähnlichartiger Schnittstellen zur Einbindung der Akteninhalte aus unterschiedlichen Quellsystemen in das Einweiserportal dar. Laut der Skalierbarkeitsanforderung R2.3 muss das Einweiserportal um die Dokumententypen zusätzlicher Organisationseinheiten des Universitätsklinikums erweitert werden können. Dazu sind weitere, zu den Schnittstellen S3 bis S5 ähnliche Schnittstellen erforderlich. Sie müssen dem System dynamisch hinzugefügt werden können. Die daraus abgeleitete Erweiterbarkeitsanforderung R4.1 verlangt, dass das Einweiserportal um zusätzliche Quellsysteme, unabhängig von der dort verwendeten Technologie, erweitert werden können muss. Kombiniert man alle diese Anforderungen mit der Architekturentscheidung, den Zugriff auf die Funktionalität mit dem Multi-Channel-Access-Provider-Pattern nach außen hin durch unterschiedliche Schnittstellen und Standards konsumierbar zu gestalten, so folgt daraus, dass die Basisschicht des „Multi Channel Access Providers“ in Form einer „Adapter Registry“ (siehe Anhang S. 254) zu gestalten ist. Das Adapter-Registry-Pattern ermöglicht den vereinheitlichten Zugriff auf eine Menge von Systemen mit ähnlicher Funktionalität, indem es Unterschiede z. B. aus den Zugriffsmechanismen in spezifischen Adapterimplementierungen kapselt. Außerdem verbirgt es, wenn gewünscht, auch die Anzahl der durch Adapter eingebundenen Systeme, da es deren Funktionalität über eine zentrale Schnittstelle gebündelt zur Verfügung stellt. Auch das Adapter-Registry-Pattern selbst ist intern in Schichten gegliedert. Es umfasst eine Registry-Schicht und eine Adapter-Schicht (vgl. Anhang S. 254, Beziehung Adapter-Registry zu Schichtenarchitektur). Durch die Anwendung der Patterns „Adapter-Registry“ und „Multi-Channel Access Provider“ wird der Einsatz des Patterns

„Schichtenarchitektur“ (siehe Anhang S. 543) zwangsläufig. Es fördert die Schaffung einer klaren Struktur in der Softwarearchitektur des Einweiserportals und unterstützt die Zuordnung der enthaltenen logischen Komponenten in die klare Gliederung des zu definierenden Schichtenmodells.

Unter Berücksichtigung des System-Kontext-Diagramms der „Inter System Interface Requirements“ und den beiden ausgewählten Patterns „Multi-Channel Access Provider“ und „Adapter-Registry“ lassen sich die folgenden Schichten identifizieren. Die Schnittstellenschicht und die Adapterschicht sind von der Vereinigung der beiden Patterns nicht direkt betroffen und können deshalb unverändert beibehalten werden. Die Basisschicht des „Multi-Channel Access Providers“ bildet durch die Kombination der beiden Patterns mit der Registry-Schicht der Adapter-Registry eine logische Einheit. Diese Einheit wird fortfolgend als Vereinigungsschicht bezeichnet, da sie der Zusammenführung oder Vereinigung der durch die Adapter ermittelten Akteninhalte zu einer kohärenten Krankenakte, verwendbar über eine zentrale Schnittstelle dient. Deshalb resultieren aus den beiden Patterns „Adapter-Registry“ und „Multi-Channel Access Provider“ die drei logischen Schichten Schnittstellenschicht, Vereinigungsschicht und Adapterschicht. Erweitert man dieses Modell um die notwendigen Quellsysteme, so ergibt sich für das Einweiserportal die folgende Schichtengliederung:

- Schnittstellenschicht
- Vereinigungsschicht
- Adapterschicht
- Quellsystemschicht.

Ein weiteres durch die Verwendung des Patterns „Adapter Registry“ implizit in die Softwarearchitektur des Einweiserportals eingeführtes Pattern ist das Microkernel-Architekturpattern. Seine Verwendung wird zudem durch die außerhalb des Wirkungsbereichs des Adapter-Registry-Patterns liegenden Anforderungen R6.1 und R6.2, die die Erweiterbarkeit des Einweiserportals auf beliebige Systeme für die Benutzerverwaltung und beliebige Lösungen zur Speicherung von Anwendungslogs hin fordern notwendig. Die Adapter-Registry ist eine spezielle und daher in ihrer Flexibilität auf eine spezielle Art von Erweiterungen beschränkte Form des Microkernels. Die Anforderungen R6.1 und R6.2 zeigen, dass eine Erweiterung der Adapter-Registry hin zu einer vollwertigen, auch um Komponenten zu den Aspekten *Benutzerverwaltung* und *Logging* erweiterbaren Microkernel-Architektur notwendig ist. Die Zweckmäßigkeit der Kombination von Microkernel- und Schichtenarchitektur wird z. B. in (Buschmann u. a. 2007a) auf Seite 175 beschrieben. Im Fall der Softwarearchitektur des Einweiserportals erstreckt sich der Wirkungsbereich der Microkernel-Architektur über die Schichten *Vereinigungsschicht* und *Adapterschicht*. Bedingt kann auch die Schnittstellenschicht als weiterer Bereich der Microkernel-Architektur betrachtet werden, da es sich bei den Bestandteilen dieser Schicht um Komponenten handelt, die das System des Einweiserportal um je eine zusätzliche Art von nach außen gerichteten Schnittstellen erweitert.

Die Erweiterungen des Microkernels werden allgemein Komponenten genannt (vgl. Buschmann u. a. 1996:S. 174). Sie lassen sich in die Typen *Internal Servers*, *External Servers*, *Adapters* und *Clients* unterteilen. In der Abhängigkeitsdarstellung zum Pattern „Microkernel“ (siehe Anhang S. 446) ist eine Generalisierungsbeziehung zwischen

komponentenbasierter Softwarearchitektur und Microkernel abgebildet. Sie zeigt, dass die Microkernel-Architektur aufgrund der Tatsache, dass sie selbst komponentenorientiert aufgebaut ist, eine spezielle Form der komponentenbasierten Softwarearchitektur ist. Folglich ist auch das Pattern „Komponentenbasierte Softwarearchitektur“ (siehe Anhang S. 419) Bestandteil des Architekturkonzepts des Einweiserportals.

Die Anforderung R5.1 fordert eine Benutzerschnittstelle, die keiner zusätzlichen Softwareinstallation auf dem Clientsystem bedarf. Das antizipiert bei Berücksichtigung von derzeit verbreiteten Technologien zur Entwicklung von Benutzerschnittstellen die Verwendung einer webbasierten Benutzerschnittstelle. Die bereits beschriebene Anwendung von Adapter-Registry und Microkernel führt zu einer Bündelung von Daten und Operationen einer Vielzahl von Systemen und integriert diese zu einer zentralen Schnittstelle. Alle diese Rahmenbedingungen erfordern in der Gruppe der grundlegenden Integrationsformen die Auswahl des Patterns „Information Portal“. Diese Auswahl wird zusätzlich von der Tatsache gestützt, dass die Integrationsform „Information Portal“ zwingend zur Anwendung des Pull-Prinzips (siehe Anhang S. 492) und der Client-Server-Architektur (siehe Anhang S. 318) führt. Beide Patterns sind auch mit den restlichen ausgewählten Patterns gut kombinierbar bzw. bereits notwendiger Bestandteil dieser Patterns.

Bei der Gestaltung der Softwarearchitektur des Einweiserportals kommt das Pull-Prinzip vom Zugriff mittels Schnittstellenschicht durch alle darunterliegenden Schichten hindurch durchgehend zum Einsatz. Das Prinzip, dass die Auslieferung von Daten durch eine gezielte Anfrage ausgelöst wird, bestimmt die Form der Kommunikation, aber auch die Verteilung der Rollen innerhalb des Einweiserportals. Die anfragende Komponente nimmt grundsätzlich die Rolle des Clients und die antwortende Komponente die Rolle des Servers ein.

Wie schon bei der Auswahl des Patterns „Komponentenbasierte Softwarearchitektur“ erläutert sind die Bestandteile der Schichtenarchitektur bzw. die Erweiterungsbestandteile des Microkernels in Komponenten organisiert. Die Schichtenarchitektur erfordert an den Übergängen zwischen den Schichten eine klar definierte öffentliche Schnittstelle, die die Interna der Schicht vor der darüberliegenden Schicht verbirgt, während sie Operationen mit verständlicher Semantik anbietet. Um diese Eigenschaft zu erreichen, ist es nützlich, das Pattern „Fassade“ (siehe Anhang S. 372) zu verwenden. Im Einweiserportal wird es z. B. zur Gestaltung der Schnittstellen des „Multi-Channel Access Providers“ eingesetzt, die nicht das Verhalten bestehender Standards adaptieren, sondern frei gestaltet werden können. Es ist eng mit dem Adapter-Pattern verwandt, das für Schnittstellen verwendet wird, die das Verhalten bestehender Schnittstellen anderer Systeme oder Standards adaptieren. Das Adapter-Pattern findet, wie bereits erwähnt, in der Adapter-Schicht der Adapter-Registry Verwendung, um die Integration oder Anbindung von bestehenden Systemen, die unterschiedliche Schnittstellen unterstützen, an das Einweiserportal zu gewährleisten. Außerdem kommt das Adapter-Pattern in den zusätzlichen Komponenten des Microkernels zur Anwendung. Adaptoren stellen in beiden Fällen eine Verbindung zwischen den innerhalb des Einweiserportals standardisierten, eindeutig definierten Schnittstellen und den verschiedenen Schnittstellenstandards der bestehenden Systeme her und tragen dadurch zur Erfüllung der Anforderungen R2.3, R6.1 und R6.2 bei.

Das Pattern „Component Configurator“ (siehe Anhang S. 325) dient der Entkopplung zwischen dem Interface und der Implementierung von Komponenten und bietet einen Mechanismus zur Rekonfiguration bzw. zum Austausch von Komponenten zur Laufzeit des Systems, ohne dass hierzu das die Komponente nutzende System gestoppt oder neu gestartet werden muss (vgl. Buschmann u. a. 2007a:S. 490). Von den bisher in diesem Kapitel für die Softwarearchitektur des Einweiserportals ausgewählten Patterns beinhalten die Patterns „Adapter Registry“ und „Microkernel“ eine nahezu zwangsläufige Verwendung des Patterns „Component Configurator“. Bei der Adapter-Registry ist die enthaltene Umsetzung des Component-Configurator-Patterns der Hauptbestandteil der Registry-Schicht. Zusätzlich sind auch die Proxys zur Integration beliebiger Benutzerverwaltungs- und Logging-Systeme als „Component Configurator“ realisiert, um einen reibungslosen und einfachen Austausch der dahinter befindlichen Komponenten zu gewährleisten. Dadurch wird auch der verbliebene Aspekt der Austauschbarkeit aus den Anforderungen R2.3, R4.1, R6.1 und R6.2 erfüllt.

Aufgrund der im Rahmen der „Inter System Interface Requirements“ identifizierten Schnittstellen der Subsysteme und der gewählten Rollenverteilung in Client- und Server-Komponenten ist das Pattern „Remote Procedure Invocation“ (siehe Anhang S. 509) notwendiger Bestandteil der Softwarearchitektur des Einweiserportals. Es gehört beispielsweise zwangsläufig zu den identifizierten Schnittstellen zum SAP System und dem Easy-Dokumentenarchiv. In seiner Gestaltung auf Designebene enthält es die Patterns „Lookup“ (siehe Anhang S. 429), „Remote Proxy“ (siehe Anhang S. 512) und „Broker“ (siehe Anhang S. 300).

Neben den behandelten Bestandteilen ist zur Planung der vollständigen Architektur des Einweiserportals auch die Einbeziehung der Aspekte *Sicherheit* und *Zuverlässigkeit* erforderlich. Die dazu nötigen Anforderungen sind in den Kapiteln 6.3.3 und 6.3.4 dokumentiert. Sie dienen als Grundlage für die Auswahl der Patterns der Schwerpunkte *Sicherheit* und *Zuverlässigkeit* in den Ebenen *Architektur* und *Design*. Die ausgewählten Sicherheitspatterns betreffen sowohl die Netzwerk- als auch die Softwarearchitektur des Einweiserportals. Im Zuge der Erhebung der speziellen Rahmenbedingungen für die Sicherheit des Einweiserportals (Kapitel 6.3.3, S. 225) kommt das Pattern „Enterprise Partner Communication“ zur Anwendung. Es dient der Beschreibung der Kommunikation mit Partnereinrichtungen. Für das Einweiserportal ist dazu auf S. 225 ein Verbindungsaufbau über eine am Universitätsklinikum Regensburg betriebene VPN-Gateway beschrieben. Die VPN-Lösung kann als bereits existierender Infrastrukturbestandteil, der zur Umsetzung des Zugriffs auf das Einweiserportal verwendet werden muss, betrachtet werden. Sie kann somit als einziger Zugriffsweg für die Nutzung der Dienste des Einweiserportals dienen. Aus diesem Grund ist für die Gestaltung der netzwerkseitigen Sicherheitsarchitektur das Pattern „Single Access Point“ (siehe Anhang S. 566) zu verwenden. Das Pattern „Single Access Point“ ist allerdings auch geeignet um, eine Ebene tiefer, die Sicherheitsarchitektur des Einweiserportals selbst zu beschreiben. Durch die Wahl des Patterns „Adapter Registry“, das den Zugriff auf eine Vielzahl von Quellsystemen durch die in seiner Registry enthaltene Sammlung von Adaptoren bündelt, ist auch in der Softwarearchitektur des Portals selbst ein zentraler Bereich definiert, durch den alle Portalzugriffe zur Ermittlung von Daten aus den Quellsystemen zwangsläufig geleitet werden. Beide Zugriffspunkte setzen an unterschiedlichen Stellen des Systems an, dienen aber dem gleichen Zweck, der Durchsetzung

von Sicherheitsregeln an zentraler Stelle, die für alle beteiligten Komponenten des Systems gültig sind.

Der „Single Access Point“ VPN-Gateway dient dabei als „Front Door“ (siehe Anhang S. 391) für den Zugriff auf die Weboberfläche des Einweiserportals und ist deshalb als „Protection Reverse Proxy“ (siehe Anhang S. 481) gestaltet. Die Zweckmäßigkeit dieser Kombination wird u. a. in (Schumacher u. a. 2005) S. 286 und S. 473 ff. beschrieben. Die Kombinierbarkeit der drei Patterns „Single Access Point“, „Front Door“ und „Protection Reverse Proxy“ ist im Anhang durch die Beziehungen zwischen den genannten Patterns dokumentiert und im Beziehungsdiagramm zu „Front Door“ (siehe Anhang S. 391) besonders gut erkennbar. Die Anwendung dieser Pattern-Kombination führt beim Einweiserportal dazu, dass kein direkter Netzwerkverkehr zwischen dem Kliniknetz und dem Internet stattfindet. Stattdessen fungiert der „Protection Reverse Proxy“ der VPN-Lösung als Stellvertreter für den Webserver des Einweiserportals und liefert die Oberfläche an dessen Stelle an den Browser des zuverlässig zu authentifizierenden Benutzers aus.

Die Anforderungen R8.3 und R8.4 zeigen, dass das Einweiserportal seine Inhalte nur abhängig von individuellen Berechtigungen der beteiligten Benutzer zugängig machen darf. Zur Durchsetzung dieser Berechtigungen gibt es in der Gruppe „Type of Access“ zwei grundlegende Lösungsansätze. Aus diesen Lösungsansätzen ist, abhängig vom Anwendungsfall, der geeignete Ansatz auszuwählen.

Die Menge der Dokumente, die in den Quellsystemen des Einweiserportals prinzipiell verfügbar ist, ist verglichen mit der Menge von Dokumenten, die ein einzelner Benutzer einsehen und verwenden darf, relativ groß. Würde das Pattern „Full Access with Errors“ (siehe Anhang S. 393) verwendet, so wäre es für den Benutzer relativ schwer, in der Menge der angezeigten Dokumente die für ihn zugängigen Dokumente zu ermitteln. Statt mit dem System arbeiten zu können, würde er durch eine Vielzahl von Meldungen über Berechtigungsfehler von seiner Arbeit abgehalten werden. Außerdem kann bereits die Information über die Existenz eines Dokuments oder den Titel eines Dokuments als personenbezogenes Datum betrachtet werden und so datenschutzrelevant sein. Aus diesen Gründen ist das Pattern „Limited Access“ (siehe Anhang S. 425) der geeignetere Ansatz. Es führt zu einem System, das alle Funktionen und Inhalte verbirgt, zu denen der angemeldete Benutzer keinen Zugang hat.

Aus der Verwendung des Patterns „Limited Access“ resultieren drei Abhängigkeiten (siehe Anhang S. 425, Abbildung der Beziehungen des Patterns „Limited Access“). Um vor jedem Benutzer genau die Menge der Funktionen und Inhalte zu verbergen, zu deren Verwendung er nicht berechtigt ist, ist es erforderlich Benutzer eindeutig und zuverlässig zu identifizieren. Deshalb besteht eine Beziehung zwischen dem Pattern „Limited Access“ und der Gruppe „Authentication“. Um die Einschränkungen des Zugriffs zuverlässig zu gewährleisten, sind im Softwaredesign verschiedene von Mechanismen für die Berechtigungssteuerung anzuwenden. Aus diesem Grund besteht eine Beziehung zur Gruppe „Access Control and Authorization“. Die letzte der drei ausgehenden Beziehungen ist eine Beziehung zum Pattern „Checkpoint“ (siehe Anhang S. 310). Diese Kombination wird empfohlen (siehe Schumacher u. a. 2005:S. 319) um die Identifikation und Authentifizierung sowie die Autorisierung zentral

umzusetzen. Das Pattern „Checkpoint“ wird auch durch das bereits verwendete Pattern „Single Access Point“ referenziert und sollte deshalb als wahrscheinlich sinnvoller Bestandteil der Architektur des Einweiserportals betrachtet werden.

In der bis zu diesem Punkt definierten Architektur des Einweiserportals sind zwei Bereiche für die Verwendung eines Checkpoints geeignet. Bezogen auf die Authentifizierung und rollenbasierte Zugriffsregelung für die Rollen *Benutzer* und *Administrator* wird die VPN-Lösung als Checkpoint eingesetzt. Bezogen auf die Durchsetzung von Berechtigungen auf die Dokumente (vgl. R8.3) aus den Quellsystemen ist hingegen kein Checkpoint implementiert. Hier ist es erforderlich, die Mechanismen direkt in den Adaptern umzusetzen, um den verschiedenen Eigenschaften der unterschiedlichen Quellsysteme und der unterschiedlichen auswertbaren Metadaten zu den Dokumenten Rechnung zu tragen. Die hierzu in den Adaptern verwendeten Patterns sind das File-Authorization-Pattern (siehe Anhang S. 381) und das Metadata-based-Access-Control-Pattern (siehe Anhang S. 445). Sie dienen der Steuerung der Zugriffsberechtigungen für einzelne Dokumente auf der Basis der zu dem jeweiligen Dokument erfassten Metadaten.

Die Patterns „Limited Access“ und „Checkpoint“ verweisen auf die Gruppen „Access Control or Authorization“ und „Authentication“. In den Anforderungen zum Einweiserportal (siehe Kapitel 6.3.4) sind „Access Control Requirements“ und „I&A Requirements“ erfasst. Aus diesem Grund wird die durch die genannten Patterns notwendige Auswahl von Inhalten der beiden Gruppen, ausgehend von den erfassten Anforderungen, durchgeführt. Von dort führt der Pfad der Beziehungen zum Pattern „Automated I&A Design Alternatives“ (siehe Anhang S. 291). Es beschreibt eine Sammlung verschiedener, alternativ zueinander oder auch gemeinsam verwendbarer automatischen Identifikations- und Authentifizierungsmechanismen, wie z. B. Passwörter, Biometrie oder Hardware-Tokens (siehe Schumacher u. a. 2005:S. 207, oben).

Die Anforderung R11.1 beschreibt, dass eine Anmeldung für die Benutzer des Einweiserportals nur möglich sein darf, wenn diese ein aus mehreren Faktoren bestehendes Authentifizierungsverfahren durchlaufen haben. Das bedeutet, dass aus der Gruppe „Anzahl der Authentifizierungskriterien“ das Pattern „Mehrfaktorige Authentifizierung“ anzuwenden ist. Es bedingt, dass aus der Menge der im Pattern „Automated I&A Design Alternatives“ verfügbaren Verfahren zwei miteinander kombinierbare ausgewählt werden müssen. Für das Einweiserportal ist eine Kombination aus Wissen und Besitz, wie in „Security Patterns“ (Schumacher u. a. 2005) auf S. 208 beschrieben, sinnvoll, da aufgrund der überschaubaren Benutzerzahl das Verteilen von Hardwaretokens finanzierbar und im Vergleich zu den biometrischen Alternativen einfacher realisierbar ist. Außerdem wird dieses Verfahren von der bereits existierenden VPN-Lösung unterstützt.

Die Abbildung 84 zeigt die in diesem Kapitel durch schrittweise Patternauswahl und systematische Kombination entwickelte Software- und Systemarchitektur des Einweiserportals. Aus Gründen der Übersichtlichkeit sind einige der untergeordnet eingesetzten Designpatterns nicht mit dargestellt. Ihre Einordnung ist daher nur dem Text des vorliegenden Kapitels zu entnehmen und für ein grundlegendes Verständnis des Modells nicht zwangsläufig erforderlich. Sie dienen ausschließlich der Qualität der konkreten softwaretechnischen Umsetzung des Einweiserportals.

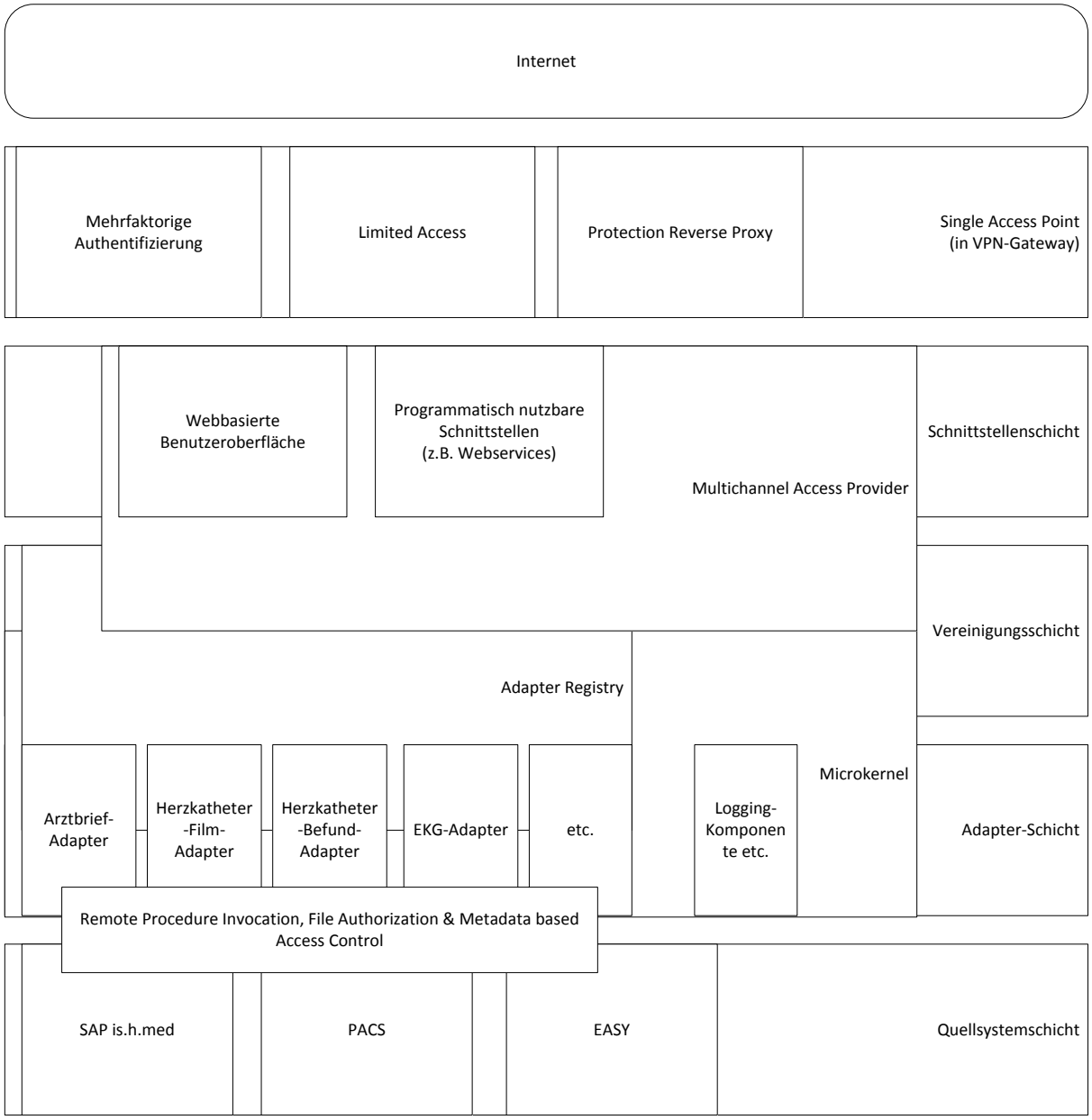



Abbildung 84: Architekturdarstellung des Einweiserportals

In absoluten Zahlen betrachtet, kommen im Einweiserportal mit 56 eingesetzten Patterns ca. 25% der Patterns aus dem Kernbestandteil der Pattern-Sprache zum Einsatz. Bezogen auf die Ebenen und Schwerpunkte aus dem Strukturkonzept lassen sich die Anteile der eingesetzten Patterns in Form einer Matrix (siehe Tabelle 17) darstellen.

| **Schwerpunkt** | **Anforderungsanalyse** | **%** | **Architektur** | **%** | **Design** | **%** |
|---|---|---|---|---|---|---|
| Flexibilität | 8 von 11 | 73 | 11 von 28 | 39 | 6 von 45 | 13 |
| Kommunikation | 6 von 9 | 67 | 7 von 20 | 35 | 0 von 4 | 0 |
| Sicherheit | 13 von 16 | 81 | 8 von 27 | 30 | 2 von 9 | 22 |
| Zuverlässigkeit | 6 von 12 | 50 | 2 von 18 | 11 | 1 von 59 | 2 |

Tabelle 17: Anteile der verwendeten Patterns nach Ebenen und Schwerpunkten

Tabelle 17 zeigt einen grundsätzlich relativ gleichmäßig über die Schwerpunkte verteilten Einsatz der Patterns. Der geringere Anteil des Schwerpunkts *Zuverlässigkeit* lässt sich dadurch erklären, dass aufgrund der Rahmenbedingungen bei der Systementwicklung nur wenige Anforderungen bezüglich benötigter Zuverlässigkeits-Eigenschaften definiert sind. Der Aspekt *Zuverlässigkeit* spielt aus diesem Grund auch in den Ebenen *Architektur* und *Design* nur eine untergeordnete Rolle.

## 6.5. Ergebnis der Umsetzung

Das in Kapitel 6.4 beschriebene Konzept wurde von Mitte 2007 bis Anfang 2009 technisch umgesetzt. Ergebnis dieser Umsetzung ist das Softwaresystem des Einweiserportals. Es ist als eine JEE5[21]-Applikation realisiert und wird auf einem JBoss Application Server[22] betrieben. Das Portal wird seit Mitte 2008 von einem kleinen Anwenderkreis, bestehend aus vier Einweisern des Universitätsklinikums Regensburg verwendet. Des Zweck des Einweiserportals ist, wie in Kapitel 6.2 beschrieben, die prototypische Erprobung der Bereitstellung von Befunddokumenten und zugehörigen Informationen für berechtigte Personen außerhalb des Universitätsklinikums.

Die Abbildung 85 zeigt den Zustand der Benutzeroberfläche des Einweiserportals, den ein Benutzer nach der erfolgreichen Anmeldung am System erhält.


Abbildung 85: Benutzeroberfläche des Einweiserportals mit Testdaten

Bezüglich seines Interaktionskonzepts sieht das Einweiserportal zwei grundlegende Anwendungsfälle vor. Einerseits unterstützt es mit der Darstellung aktueller Dokumente (siehe Abbildung 85) einen Ansatz der Dokumentenauswahl, der auf einer Liste der zuletzt für den aktuell angemeldeten Benutzer freigegebenen Dokumente basiert. Andererseits unterstützt es, ausgehend von einer Suche nach Patienten, eine Dokumentenauswahl aus der Gesamtmenge der für den aktuell angemeldeten Benutzer freigegebenen Dokumente zu einem Patienten. Die Komplexität der darunterliegenden Schichten wird durch die vergleichsweise schlicht gestaltete Benutzeroberfläche vor dem Endanwender verborgen. Die Freigabe von Dokumenten ist direkt und für den Benutzer transparent in den Prozess der Dokumentenerstellung im SAP-System integriert.

[21] Java Enterprise Edition 5, http://jcp.org/aboutJava/communityprocess/final/jsr244/index.html
[22] JBoss Application Server, http://www.jboss.org/jbossas

Aus dem seit Mitte 2008 stattfindenden Betrieb des Systems sind Erkenntnisse über die Eigenschaften des Einweiserportals entstanden. Da von Mitte 2008 bis Anfang 2009 das System noch intensiv weiterentwickelt wurde, ist dieser Zeitraum als nicht repräsentativ aus der Betrachtung der Zuverlässigkeitseigenschaften des Einweiserportals auszuschließen. Über die Zeitdauer von Anfang 2009 bis 2012 hinweg hat sich das System bezüglich seines Laufzeitverhaltens als stabil und bezüglich seines Berechtigungssystems und der Fehlerbehandlung als zuverlässig erwiesen. Seit der Fertigstellung der Software sind keine Fehlfunktionen in diesen Bereichen zu verzeichnen. Auftretende und protokollierte Fehler beschränken sich auf typische Verbindungsfehler, die aus der temporären Nichtverfügbarkeit einzelner Quellsysteme resultieren. Im Betrachtungszeitraum sind solche Fehler ungeplant nur in geringer Zahl und ausschließlich für die Verbindung zum Herzkatheter-PACS aufgetreten.

Die auf der Seite 213 durch die zu erwartenden Veränderungen V1 bis V4 dokumentierten Anforderungen an die Flexibilität werden erfüllt. Die Erweiterbarkeit des Portals um zusätzliche Einrichtungen und Nutzer (V1) ist durch die Erweiterung des Nutzerkreises von 2 externen Nutzern auf 4 externe Nutzer sowie durch die Anlage von einer größeren Zahl von Testbenutzern eindeutig belegbar. Im Zeitraum von 2009 bis 2012 sind am Universitätsklinikum zwei umfangreiche Upgrades des SAP is.h.med Systems vorgenommen worden. Trotzdem ist das Einweiserportal weiterhin fähig, mit diesem System zu kommunizieren. Das belegt, dass auch Veränderungen der Art V2 im Architekturkonzept des Einweiserportals berücksichtigt sind. Zu Testzwecken wurde Anfang 2009 ein zusätzlicher Adapter für eine Dokumentenart aus dem Bereich der Augenheilkunde entwickelt. Dieser Adapter wird zwar nicht produktiv eingesetzt, er belegt aber, dass zusätzliche Inhalte einfach zur Verfügung gestellt werden können (V3). Während der späten Entwicklungsphase wurden im Rahmen der Weiterentwicklung des Netzwerkkonzepts im Kliniknetz die IP-Adressbereiche verschiedener Subnetze umgestellt. Das machte eine Rekonfiguration der verschiedenen Verbindungen der Adapter notwendig. Dabei gab es keine Probleme. Somit ist auch die zu erwartende Veränderung V4 nachweislich durch das System unterstützt. Deshalb kann festgehalten werden, dass das System bezüglich seiner Flexibilität und Anpassbarkeit die notwendigen Voraussetzungen erfüllt.

Trotzdem weist das Einweiserportal eine niedrige Nutzungshäufigkeit durch die vier registrierten Benutzer auf. Laut den Aussagen der Benutzer ist das auf die fehlende Integration in die Primärsysteme der Einrichtungen außerhalb des Universitätsklinikums zu erklären. Der Wechsel zwischen den Oberflächen zweier Systeme, dem Praxis- oder Krankenhausinformationssystem der jeweiligen Einrichtung und dem Einweiserportal, verursacht zusätzlichen Zeitaufwand, ist unbequem und wird deshalb nur selten durchgeführt. Eine bessere clientseitige Integration scheitert aber am Fehlen geeigneter, für alle beteiligten Produkte verfügbarer Standards, wie die Diplomarbeit von Zorneck (Zorneck 2009) zeigt. Die beiden zum Zeitpunkt der Implementierung für Praxissoftware verfügbaren Standards D2D (Zorneck 2009:S. 33 ff.) und VCS (Zorneck 2009:S. 38 ff.) sind nur in wenigen Produkten verfügbar und setzen auf die asynchrone Übertragung aller für die Nutzer der jeweiligen Installation potenziell interessanten Dokumente. Diese Art der Kommunikation erzeugt eine große Menge unnötiger Daten im System des Empfängers. Sie sorgt für Redundanz, macht eine nachträgliche, z. B. durch den expliziten Wunsch des Patienten bedingte Rücknahme einer Freigabe technisch unmöglich und widerspricht außerdem dem grundlegenden

Kommunikationsmodell des Einweiserportals. Außerdem betreibt nur die Hälfte der relevanten Nutzer klassische Praxissoftware. Für die Krankenhausinformationssysteme der anderen Nutzer sind die beiden genannten Standards nicht vorgesehen. Aus diesen Gründen ist ihre Umsetzung im Einweiserportal nicht sinnvoll.

Im Gegensatz zur schlecht in die Arbeitsumgebung der externen Anwender integrierten externen Schnittstelle des Einweiserportals wird die transparent in die Dokumentenerstellung integrierte Freigabefunktionalität fortlaufend und für alle Dokumente der integrierten Typen verwendet. Diese Tatsache unterstreicht die Aussagen der Anwender, dass vermehrte Nutzung in erster Linie durch eine nahtlose Integration mit den dort bereits bestehenden Verfahren erreicht werden kann. Sie darf allerdings nicht als Beweis für die Richtigkeit dieser Annahme angesehen werden.

Insgesamt betrachtet zeigt das Beispiel *Einweiserportal*, wie durch die Nutzung des vorgestellten patternbasierten Ansatzes Softwaresysteme für verteilte Krankenakten entwickelt werden können, die auch nichtfunktionale Anforderungen aus den Bereichen *Flexibilität*, *Sicherheit* und *Zuverlässigkeit* problemadäquat erfüllen.

# 7. Fazit

In der vorliegenden Arbeit werden in den Kapiteln 2 und 3 die Rahmenbedingungen, die für eine allgemeingültige Methode und Methodik zur Entwicklung von verteilten Krankenakten gelten, ermittelt und detailliert dargestellt. Ausgehend von diesen Rahmenbedingungen zeigt Kapitel 4, dass das Konzept der Patterns, einschließlich des zugehörigen Konzepts der Pattern-Sprache, eine besonders geeignete Basis für die Entwicklung einer solchen Methode ist. Patterns sind sowohl in der praktischen Softwareentwicklung als auch im wissenschaftlichen Bereich etabliert. Deshalb eignen sie sich sowohl dazu, die Entwicklung von verteilten Krankenakten zu vereinfachen, als auch dazu, die Kommunizierbarkeit der Inhalte zwischen Wissenschaftlern und Praktikern *(vgl. Kapitel 3.1)* zu verbessern *(siehe Kapitel 3.2.3, S. 113)*.

Aufgrund der ermittelten Rahmenbedingungen ergeben sich nach ausklammern inhaltlicher Planungsaspekte des Domänenmodells, die Kommunikation, Flexibilität, Sicherheit und Zuverlässigkeit als besondere technische Schwerpunkte bei der Entwicklung verteilter Krankenakten. Sie gewinnen durch den Aspekt der Verteiltheit im Vergleich zu nicht verteilt genutzten oder realisierten elektronischen Krankenakten erheblich an Bedeutung *(siehe Kapitel 4.2.2 und 2.4.2)*. Durch die notwendige Verteilung wird die Kommunikationsfähigkeit der einzelnen Bestandteile unerlässlich. Kommunikation kann wiederum ohne geeignete Schutzmaßnahmen die Sicherheit und Zuverlässigkeit von Systemen negativ beeinflussen. Außerdem ist durch die möglicherweise große Zahl und Vielfalt der beteiligten Kommunikanten auch eine erhöhte Flexibilität bezüglich verschiedener Faktoren erforderlich.

Kapitel 5 enthält die Beschreibung einer Sammlung von 220 Patterns, die die Umsetzung von kommunikationsfähigen, flexibel integrierbaren sowie zuverlässigen und sicheren verteilten Krankenakten unterstützen. Die Sammlung ist nach den Prinzipien einer Pattern-Sprache aufgebaut und enthält ein Strukturkonzept zur Abbildung von Gliederungselementen und Beziehungen, das durch das in Kapitel 4.3 beschriebene Metamodell auch in einem Softwaresystem zur Verwaltung der Patternsammlung abgebildet werden kann. Die Patternsammlung erfüllt die durch Christopher Alexander für Pattern-Sprachen definierten Vollständigkeitsanforderungen nicht in vollem Umfang, erreicht allerdings einen für die praktische Verwendung der Sammlung bereits ausreichenden Grad an Vollständigkeit und wird deshalb als der Kernbestandteil der Pattern-Sprache für die Entwicklung verteilter Krankenakten bezeichnet.

Kapitel 6 zeigt, wie mit dieser Sammlung systematisch Softwaresysteme für verteilte Krankenakten entwickelt werden können. Die hierfür notwendige Methodik zur systematischen Anwendung der Pattern-Sammlung wird in Kapitel 6.1 allgemein und unabhängig von konkreten Systemen erläutert. In den Kapiteln 6.2 bis 6.5 untermauert das Beispiel eines real existierenden Systems – des Einweiserportals des Universitätsklinikums Regensburg, das im Rahmen dieser Arbeit entstanden ist – die praktische Anwendbarkeit des vorgestellten Patternsystems. Es demonstriert, wie ein systematisches Vorgehen bei der Nutzung des Kernbestandteils der Pattern-Sprache für verteilte Krankenakten aussehen kann.

Das Vorliegen von Defiziten am Einweiserportal zeigt aber auch, dass die beschriebene Methode nicht in der Lage ist, sämtliche Fehler zu vermeiden. Das Problem der fehlenden Integration mit den Praxissystemen der Einweiser entstand trotz der Nutzung des Kernbestandteils der Pattern-Sprache für verteilte Krankenakten. Fehler die durch falsche Annahmen bei der Erstellung von Anforderungen oder durch technisch unflexible Partnersysteme entstehen, können auch durch die Anwendung der vorliegenden Methode nicht ausgeschlossen werden. Bei rechtzeitig erkannten Defiziten oder einer Veränderung der Rahmenbedingungen können Probleme aber ggf. durch die Anwendung eines dafür vorgesehenen Patterns gemildert werden. So könnten die nicht eingebundenen Praxissysteme, sobald sie über eine Implementierung eines zur Einbindung geeigneten Standards verfügen, ohne tief greifende Änderungen am Einweiserportal eingebunden werden, da dort das Multi-Channel-Access-Provider-Pattern umgesetzt ist, das die Bereitstellung mehrerer Zugriffswege oder Standards unterstützt.

Mit ihrer Beschreibung in den Kapiteln 4, 5 und 6 erfüllt die vorgestellte Pattern-Sprache für verteilte Krankenakten mit ihrem Kernbestandteil als patternbasierter Ansatz zur Entwicklung verteilter Krankenakten die in Kapitel 3.2.1 auf Basis der entsprechenden Definitionen von Sommerville sowie Ludewig und Lichter erarbeiteten Anforderungen an eine Methode des Softwareengineering. Sie kann deshalb als Methode des Softwareengineering bezeichnet werden. Die verschiedenen Bestandteile der Methode, die Patterns und deren Beziehungen sind im Zeitverlauf Veränderungen bezüglich ihrer Wertigkeit bzw. Gültigkeit unterworfen. Abhängig von der eingesetzten Entwicklungsplattform und den eingesetzten Frameworks kann durch Neuerungen der Einsatz bislang als notwendig erachteter Patterns an Wichtigkeit verlieren oder obsolet werden (vgl. z. B. Bien 2009). Gleichzeitig gewinnen ggf. neue Patterns an Relevanz. Deshalb muss die der Methode zugrunde liegende Pattern-Sprache stetig weiterentwickelt werden. Dazu ist kontinuierliche Forschung hinsichtlich neuer relevanter Bestandteile und Veränderungen im enthaltenen Beziehungsgeflecht erforderlich.

Um eine solche kontinuierliche Weiterentwicklung durch Forschung zu gewährleisten ist die Beteiligung einer größeren Zahl von Personen notwendig. Diese kann durch eine Weiterentwicklung des Softwaresystems zur Verwaltung des Patternsystems, hin zu einem auf die Verwendung durch eine größere Community optimierten Softwaresystem, unterstützt werden. Dazu ist es notwendig, das bestehende System um ein Benutzerkonzept, Kommentarfunktionen, eine benutzerbezogene Änderungsverfolgung zur Feststellung der Urheberschaft bei relevanten Entwicklungsschritten und eine Versionierungs- bzw. Historisierungsfunktion zu erweitern, um damit einen kontinuierlichen gemeinsamen Forschungsprozess zu unterstützen. Um die Qualität der Patterns und Beziehungen in einem solchen Prozess dauerhaft zu berücksichtigen ist eine Integration qualitätsbeschreibender Attribute in das Metamodell der Pattern-Sprache sinnvoll. Bei der Auswahl der Art der Attribute kann beispielsweise (Wurhofer u. a. 2009) als geeignete Quelle hinzugezogen werden.

Der patternbasierte Ansatz zur Entwicklung verteilter Krankenakten kann in seiner vorliegenden Form einen wesentlichen Beitrag zur Entwicklung von Softwaresystemen für verteilte Krankenakten leisten. Werden die gegen Ende des Fazits erläuterten zusätzlichen Anforderungen erfüllt und eine interessierte Community aufgebaut oder gefunden, so kann er

als Methode zur Entwicklung verteilter Krankenakten auch dauerhaft zur Vereinfachung der Entwicklung solcher Systeme und zur Erforschung der Einflussfaktoren auf die Qualität verteilter Krankenakten beitragen.

# Anhang A: Pattern- und Beziehungskatalog

## *1. Abstract Factory*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 87 bis 95

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

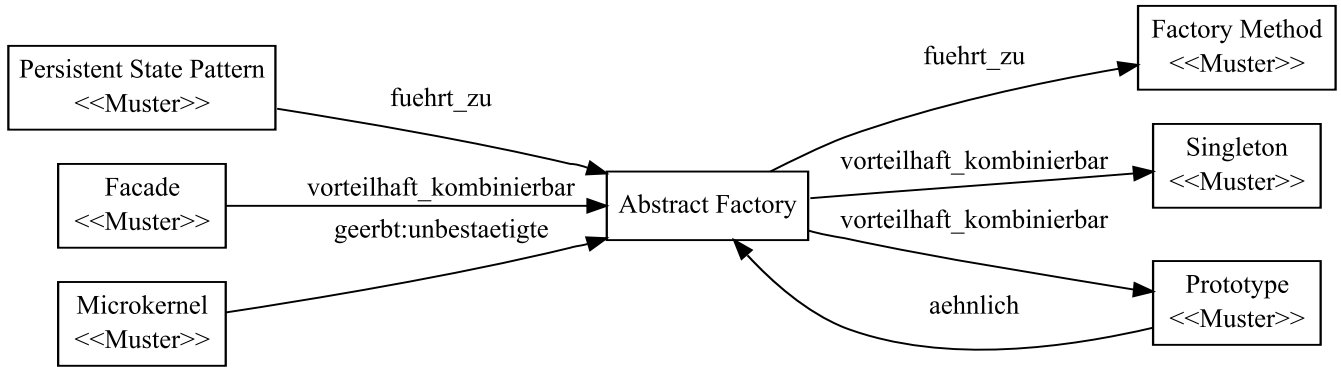
Gruppe(n)

- Instanziierung und Kontrolle der Instanzen

Kurzbeschreibung

Das Pattern "Abstract Factory" beschreibt den Weg, Methoden bereitzustellen, die Instanziierung von konkreten Klassen desselben Interfaces bzw. einer gemeinsamen Eltern-Klasse kapseln. So können abhängig von Parametern oder sonstigen Eigenschaften der Umgebung unterschiedliche implementierende Klassen mit gleicher Schnittstelle zur Verfügung gestellt werden.

Beziehungen - Grafik




Beziehungen - Detail

- **Persistent State Pattern -> Abstract Factory** : create and control a transaction lifecycle
  Das Abstract-Factory-Pattern ist als Transaction-Factory ein Bestandteil des Persistent-State-Pattern.
  Quelle: http://www.hillside.net/plop/2010/papers/saude.pdf Nr. 4.5 Structure und Nr. 4.7 Related Patterns

- **Facade -> Abstract Factory** : Schnittstelle zur Generierung von Subsystem-Objekten
  Quelle: Design Patterns; Gamma et al.; S. 193
  *Abstract Factory can be used with Facade to provide an interface for creating subsystem objects in a subsystem-independent way.*
- **Prototype -> Abstract Factory** :
  Quelle: Design Patterns; Gamma et al.; S. 119;
  *Prototype has many of the same consequences that Abstract Factory and Builder have: ...*
- **Abstract Factory -> Factory Method** : Umsetzbar unter Zuhilfenahme von
  Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Abstract Factory -> Singleton** : for single instance
  Um von einer abstrakten Factory nur eine Instanz zu generieren und diese überall verwenden zu können, wird die Kombination mit dem Singleton Muster benötigt.
  Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Abstract Factory -> Prototype** : Configure factory dynamically
  Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen
  Geerbte Beziehung: unbestaetigte_beziehung
  Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.

Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

## 2. Access Control List

Quellen

**Patterns for access control in distributed systems**

Patterns for access control in distributed systems; Delessy, Nelly and Fernandez, Eduardo B. and Larrondo-Petrie, M. M. and Wu, Jie; in Proceedings of the 14th Conference on Pattern Languages of Programs; PLOP '07; Monticello, Illinois; http://doi.acm.org/10.1145/1772070.1772074; Alternativer Download-Link: http://hillside.net/plop/2007/papers/PLoP2007_DelessyEtAl.pdf

Schwerpunkt(e)

- Sicherheit

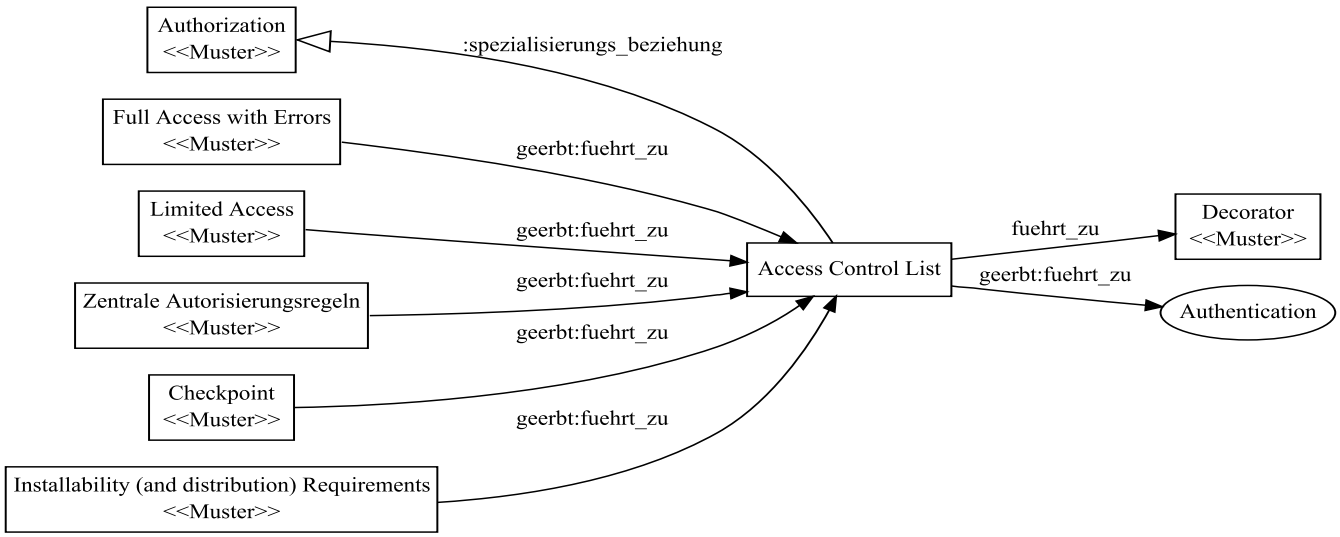
Gruppe(n)

- Access Control or Authorization

Kurzbeschreibung

Das Pattern "Access Control List" beschreibt ein Verfahren, Zugriffsrechte für je ein Objekt mittels einer Matrix mit den Dimensionen Benutzer und Recht zu verwalten.

Beziehungen - Grafik




Beziehungen - Detail

- **Access Control List -> Decorator** : can be implemented as
  Die Klassendiagramme in
  http://hillside.net/plop/2007/papers/PLoP2007_DelessyEtAl.pdf Figure 4 sowie

Gamma, Johnson, Vlissides, Design Patterns S. 177 legen nahe, dass eine Umsetzung von ACLs mittels Decorator in vielen Fällen sinnvoll ist. Außerdem bestärkt auch die folgende Aussage (aus http://hillside.net/plop/2007/papers/PLoP2007_DelessyEtAl.pdf) diese Vermutung: *"Implement the Access Matrix by associating each object with an Access Control List (ACL) that specifies which actions are allowed to be performed on the object and by which authenticated users."*

- **Full Access with Errors -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.
- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Konzepte zur Verteilung der Autorisierungsinformation lassen sich durch die Kombination von Patterns der Gruppe Access Control or Authorization umsetzen.
  Bei der Verwendung zentraler Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.
- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al., S. 287;
  *"..., a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*
- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts

notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.

## 3. Access Control Requirements

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 267 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit
- Zuverlässigkeit

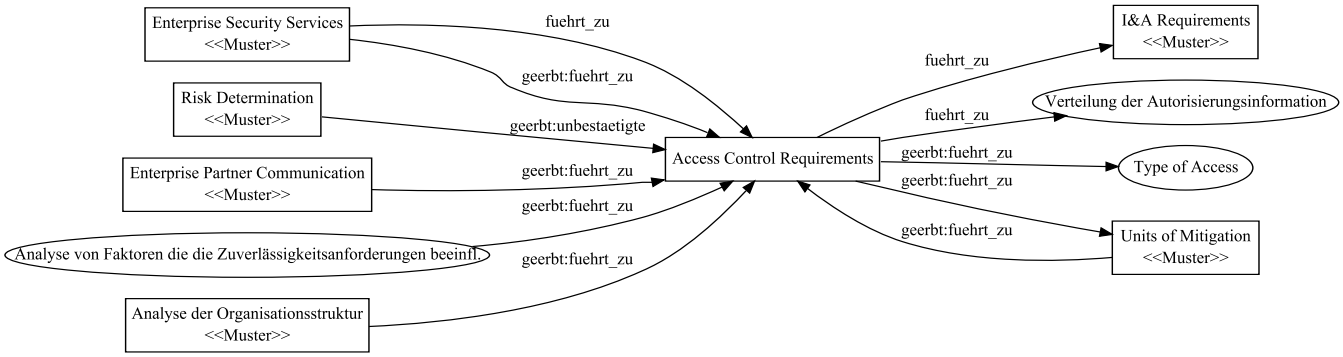
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen
- Ermittlung der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Access Control Requirements" beschreibt ein Verfahren zur Ermittlung und Definition von Anforderungen bezüglich der Einschränkung des Zugriffs auf Bestandteile eines Softwaresystems. Dazu greift es auf verallgemeinerte Anforderungstypen zurück und beschreibt einen Prozess zur systematischen Anforderungsdefinition.

Beziehungen – Grafik




Beziehungen - Detail

- **Enterprise Security Services -> Access Control Requirements** : als Autorisierungsdienst
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und*

*Sicherheitsmanagement.“* (übersetzt aus dem Englischen)
Quelle: Security Patterns, Schumacher et al.; S. 267: „*Basierend auf den Ergebnissen der Anwendung von Enterprise Security Services, ...“* (übersetzt aus dem Englischen)
Die Spezifikation von Access Control Requirements führt zu einer Feinspezifikation der für die Autorisierung spezifizierten Inhalte aus der Anwendung des Enterprise-Security-Services-Patterns.

- **Access Control Requirements -> I&A Requirements** : wenn Access Control dann auch Authentifizierung
  Wenn Access Control Mechanismen benutzt werden sollen, dann ist auch Authentifizierung notwendig. Aus diesem Grund müssen, als logische Konsequenz der Spezifikation von Access Control Requirements auch I&A Requirements spezifiziert werden.
- **Access Control Requirements -> Verteilung der Autorisierungsinformation** : Entscheidung für oder gegen die Verteilung des Autorisierungsmechanismus
  Die spezifizierten Access Control Requirements beeinflussen die Verteilbarkeit von Autorisierungsregeln und dienen als so Informationsbasis für die Auswahl des geeigneten Patterns aus der Gruppe Verteilung der Autorisierungsinformation.
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.
- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** : Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“* (übersetzt aus dem Englischen)
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche

Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.

- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** : Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** : Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.
- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units

of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## 4. Acknowledgement

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 104 ff.

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

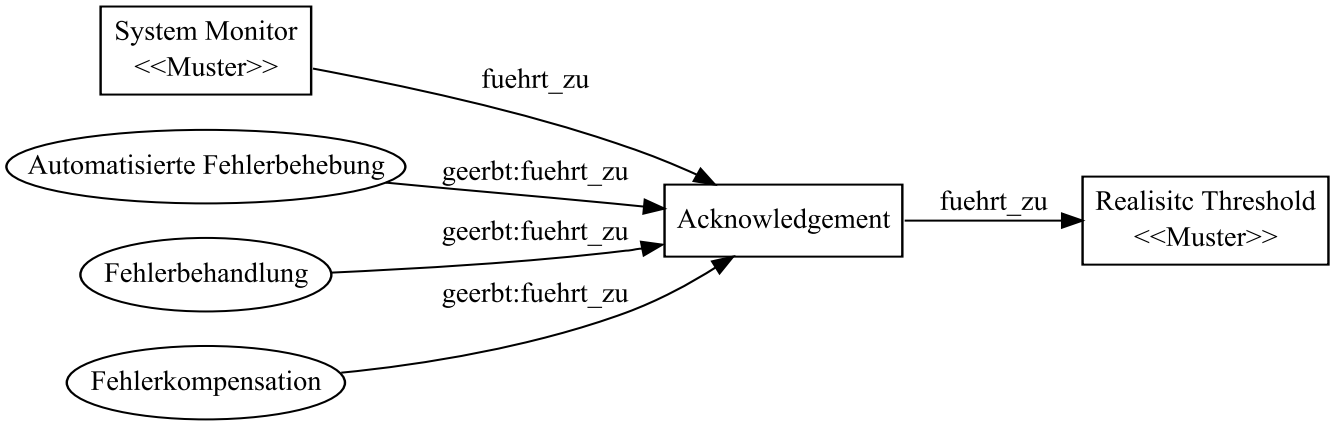
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Acknowledgement" beschreibt das Verfahren, dem Kommunikationspartner Bestätigungsnachrichten für erhaltene Nachrichten zuzustellen. Dieses Verfahren hilft dabei Verbindungs- bzw. Übertragungsfehler zu erkennen.

Beziehungen - Grafik




Beziehungen - Detail

- **System Monitor -> Acknowledgement** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Acknowledgement -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen

automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung.
  Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *5. Adapter*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 139 ff.

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

Unter dem Namen Object Adapter ab S. 438 beschrieben.

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Flexibilität

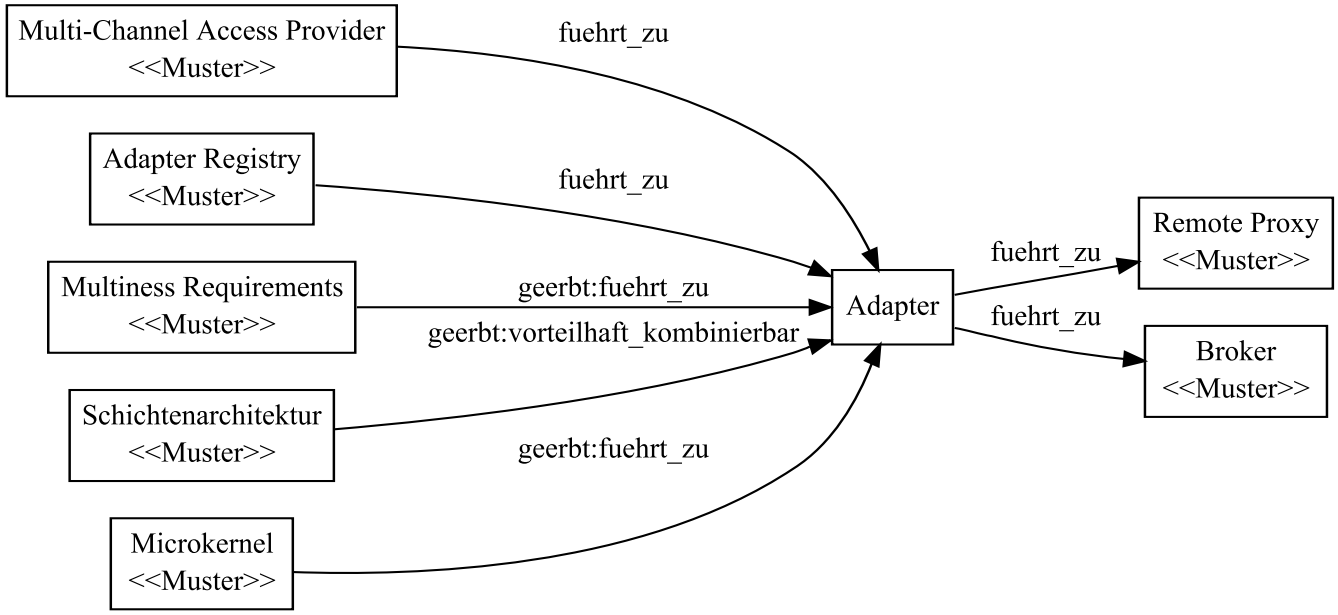
Gruppe(n)

- statische Veraenderung der Schnittstelle
- Handle Body Patterns
- Kopplung und Entkopplung

Kurzbeschreibung

Das Adapter-Pattern beschreibt einen Weg, die Schnittstelle einer bestehenden Klasse durch eine zusätzliche Klasse - die Adapter-Klasse - so zu kapseln, dass diese ursprüngliche Klasse über eine vorher bereits definierte andere Schnittstelle genutzt werden kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Multi-Channel Access Provider -> Adapter** : Adapter ist Bestandteil der Schnittstellenschicht
  Das Multi-Channel-Access-Provider-Pattern beinhaltet als Bestandteile seiner Schnittstellenschicht Adaptoren, die entsprechend dem Adapter-Pattern implementierbar sind.
- **Adapter Registry -> Adapter** : Elemente der Adapter-Schicht
  Das Adpater-Registry-Pattern ist als Schichtenarchitektur aufgebaut und besitzt eine Adapter-Schicht. Die Elemente der Adapter-Schicht implementieren sowohl das Adapter- als auch das Proxy-Pattern.
- **Adapter -> Remote Proxy** : Wenn Adaptee Remote
  Wenn das anzupassende Objekt (Adaptee) sich auf einem entfernten System befindet, ist es notwendig, die Adapter-Implementierung mit dem Remote-Proxy-Pattern oder dem Broker-Pattern zu kombinieren.
- **Adapter -> Broker** : Wenn Adaptee Remote
  Wenn das anzupassende Objekt (Adaptee) sich auf einem entfernten System befindet, ist es notwendig, die Adapter-Implementierung mit dem Remote-Proxy-Pattern oder dem Broker-Pattern zu kombinieren.
- **Multiness Requirements -> statische Veraenderung der Schnittstelle** : Indirekt: Abbildung auf Ziel-Schnittstellen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Grundsätzlich werden Muster zur statischen Veränderung der Schnittstelle verwendet um einzelne der Multiness Requirements zu erfüllen. Es ist aber zusätzlich ein größeres Pattern (Ebene Architektur) wie z. B. das Multi-Channel-Access-Provider-Pattern notwendig um die Multiness Requirements in einer konsistenten Architektur abzubilden.

- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden, um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *6. Adapter Registry*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

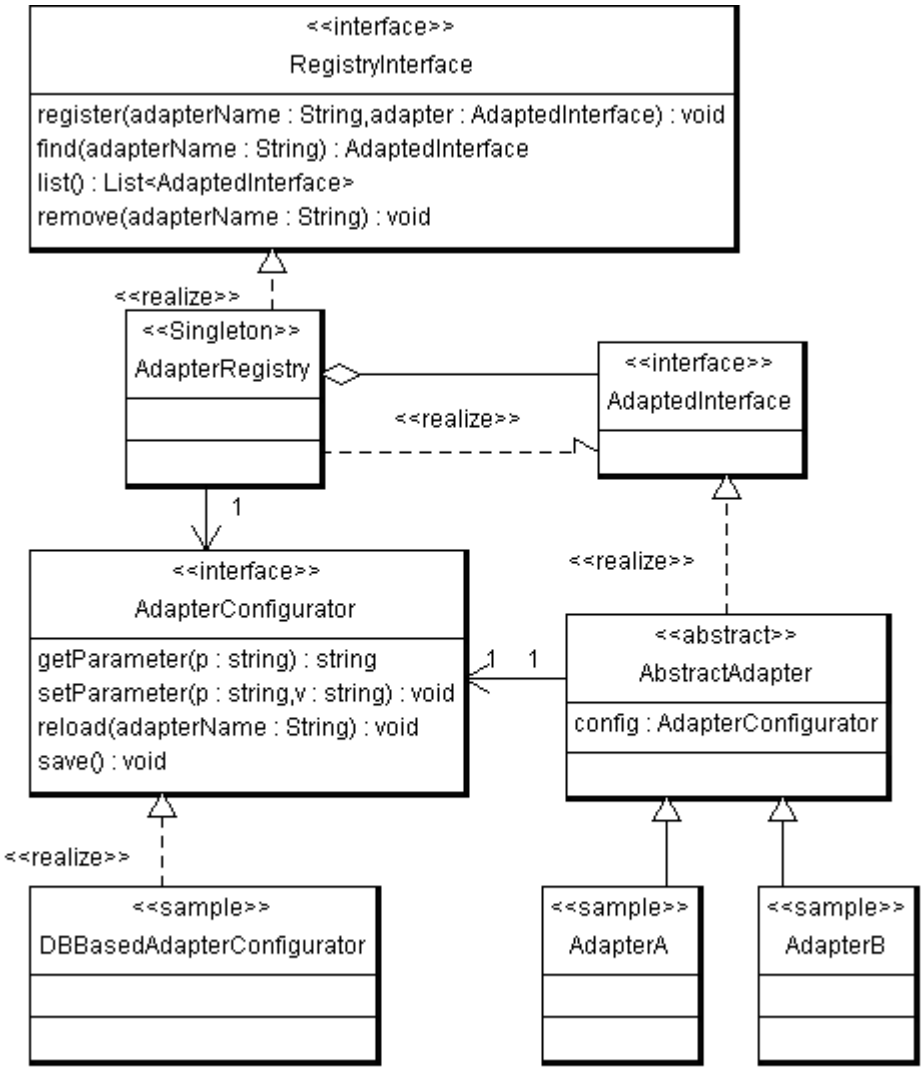
- Flexibilitaetserhoehende Architekturmuster

Kontext

Es soll ein einheitlicher Zugriffspunkt für eine Menge ähnlichartiger Systeme oder Subsysteme geschaffen werden, der die teilweise existierenden Unterschiede in den Zugriffsmechanismen verbirgt und die Erweiterbarkeit um zusätzliche Systeme oder Subsysteme möglichst einfach gestaltet.

Problem
Eine Komponente (einheitlicher Zugriffspunkt) soll für den Zugriff auf eine Vielzahl von Komponenten mit unterschiedlichen Schnittstellen (genannt Backend-Komponenten) aber ähnlichen Aufgaben eine einheitliche Schnittstelle zum Zugriff zur Verfügung stellen. Dabei müssen auch Backend-Komponenten von entfernten Systemen eingebunden werden können.

Lösung
Die Lösung des Problems besteht aus einer Architektur, die eine Vielzahl von Adaptoren für die Anbindung der verschiedenen Systeme, Subsysteme oder Komponenten (Adaptees) verwalten und deren Methoden über eine gemeinsame Schnittstelle verwendbar machen kann. Diese Architektur wird aus einer Kombination verschiedener bekannter Design Patterns aufgebaut.
**Diagramm:**



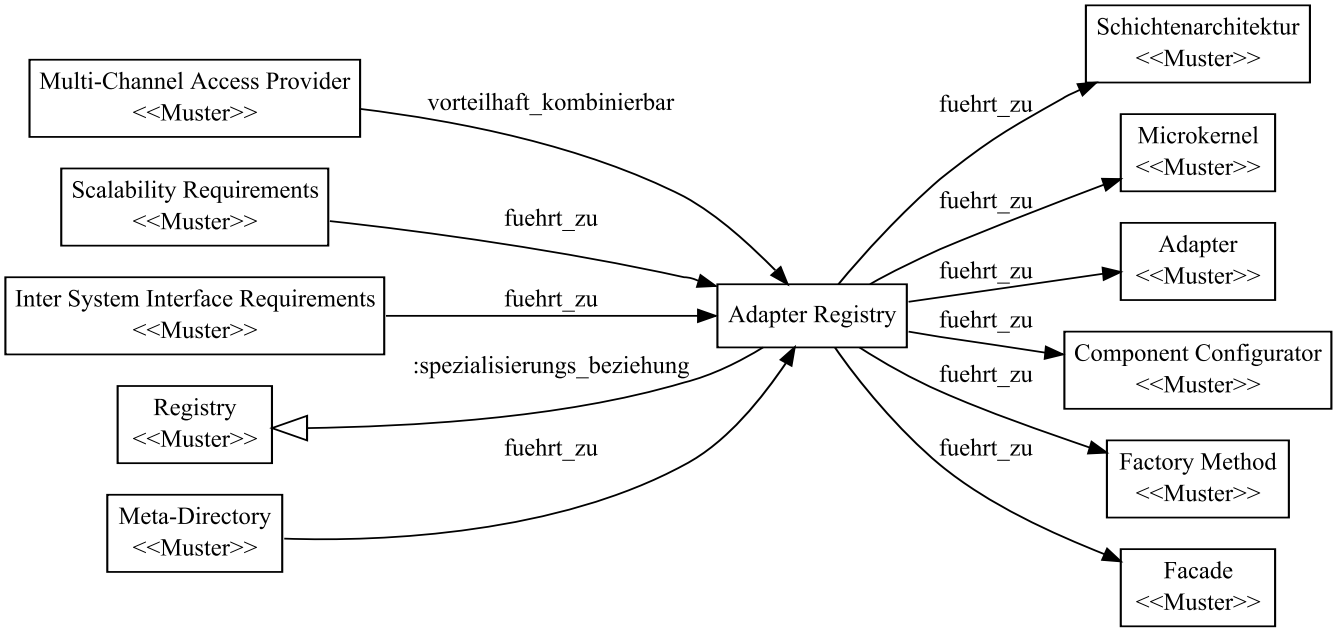

Das abgebildete UML-Klassendiagramm zeigt die allgemeine Form des Adapter-Registry-Patterns. Dabei implementiert die Registry das gleiche Interface wie die Adaptoren und erstellt so eine Fassade, die den parallelisierten Aufruf der Methoden an den verschiedenen Adaptoren verbirgt. Die Adaptoren selbst beinhalten, wenn sie mit entfernten Adaptees kommunizieren, neben dem Adapter- und Component-Configurator-Pattern zusätzlich das Remote-Proxy- oder Broker-Pattern um innerhalb der Adapter-Implementierung die Komplexität der Kommunikation mit dem entfernten Kommunikationspartner vor dem eigentlichen Adapter-Code zu verbergen. Jede Adapter-Implementierung verwendet Konfigurationsdaten die über, pro Instanz, ein Objekt der Klasse AdapterConfigurator zur Verfügung gestellt werden. Eine Adapter-Implementierung kann mit verschiedenen Konfigurationsobjekten mehrfach instanziiert und unter unterschiedlichen Namen in der Registry registriert werden. Dadurch ist es möglich unterschiedliche Instanzen gleichartiger Systeme mit gleichen Adapter-Implementierungen durch unterschiedliche Konfigurationsdaten anzubinden. Auch die Adapter-Registry selbst besitzt ein Konfigurationsobjekt der Klasse AdapterConfigurator. Sie verwaltet darin persistent die registrierten Adapter mit ihren Namen und den Namen der zugehörigen Konfigurationsobjekte.

Beispiel

Beispiele für erfolgreiche praktische Anwendung ist die im Einweiserportal des Universitätsklinikums Regensburg verbaute DocumentRegistry-Komponente. Sie entspricht exakt dem hier vorgestellten Adapter-Registry-Pattern. Eine detaillierte Beschreibung zur Architektur des Einweiserportals findet sich in Kapitel 6.

***Verwandte Lösungen:*** Eine verwandte Lösung ist die Dojo.AdapterRegistry (siehe: http://dojotoolkit.org/api/1.3/dojo/AdapterRegistry). Sie besitzt den gleichen Namen bildet aber ausschließlich den Registrierungsdienst für Adaptoren ab. Eine gemeinsame Verwendung mehrerer Adaptoren ist nicht enthalten.

Beziehungen - Grafik




Beziehungen - Detail

- **Multi-Channel Access Provider -> Adapter Registry** : als Datenquelle AdapterRegistry als Datenquelle für Multi-Channel Access Provider; Access Provider als Mittel die geschaffene vereinheitlichte Schnittstelle in einer Vielzahl von Standards zur Verfügung zu stellen.
- **Scalability Requirements -> Adapter Registry** : macht Anzahl der Backends dyn. skalierbar
  Durch die Verwendung des Adapter-Registry-Musters kann die Anzahl von (auch verschiedenartigen) Backendsystemen dynamisch erweitert oder verkleinert werden, ohne den Code des Systems zu verändern. Je nach Implementierung ist auch eine Veränderung ohne Neustart möglich.
- **Inter System Interface Requirements -> Adapter Registry** : bei vielzahl ähnlicher Subsysteme die gemeinsam genutzt werden sollen
- **Registry -> Adapter Registry** : Idee: Registrierung von Objekten
  Das Registry-Pattern wird zur Verwaltung der Adapter-Objekte in der Adapter Registry verwendet.

- **Meta-Directory -> Adapter Registry** : Umsetzbar durch
  Werden Benutzerdatenquellen mit unterschiedlichen Schnittstellen oder Standards verwendet, so kann es notwendig sein, für die Integration der Datenquellen unterschiedliche Adapter-Implementierungen zu verwenden. Der Integration Reverse Proxy kann dann unter Verwendung einer Adapter Registry implementiert werden.
- **Adapter Registry -> Schichtenarchitektur** : Interne Gliederung in Schichten
  Das Adapter-Registry-Pattern ist intern in die Schichten Registry-Schicht und Adapter-Schicht gegliedert.
- **Adapter Registry -> Microkernel** : Erweiterbarkeitsprinzip
  Es ist vorteilhaft die Registry-Schicht intern nach den Prinzipien einer Microkernel-Architektur erweiterbar zu gestalten. Die Integration der Adaptoren in die Registry-Schicht folgt diesem Prinzip bereits.
- **Adapter Registry -> Adapter** : Elemente der Adapter-Schicht
  Das Adpater-Registry-Pattern ist als Schichtenarchitektur aufgebaut und besitzt eine Adapter-Schicht. Die Elemente der Adapter-Schicht implementieren sowohl das Adapter- als auch das Proxy-Pattern.
- **Adapter Registry -> Component Configurator** : Konfigurierbarkeit von Adaptoren und Registry
  Sowohl die Adaptoren als auch die Registry werden als per Konfiguration austauschbare Komponenten umgesetzt.
- **Adapter Registry -> Factory Method** : Instanzierung der Adaptoren durch Registry
  Die Instanzierung von Adaptoren muss z. B. im Falle des Aufrufs der reload()-Methode durch die Registry erfolgen. Da zur Instanzierung der Adaptoren immer auch die Auswahl des richtigen Konfigurationsobjekts bzw. die Instanzierung des Konfigurationsobjekts mit anschließendem Laden der Konfigurationsdaten einher geht ist die Verwendung von Factory-Methoden zur Instanzierung empfehlenswert.
- **Adapter Registry -> Facade** : Adapter-Schnittstelle als Fassade der Registry-Klasse
  Die Adapter-Registry-Klasse verbirgt die Komplexität des Zugriffs auf eine Vielzahl von Adaptees mittels einer ebenso großen Zahl von Adaptoren durch die Bereitstellung aller Methoden der Adapter-Schnittstelle als Fassade der Adapter-Registry-Klasse.

## *7. Akte mit regional zentralisierter Datenspeicherung*

Englische Bezeichnung

- Health Record with regional centralized Data Storage

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Verteilungsarchitektur der Akte

### Kontext

Die Anforderungen an das zukünftige verteilte Krankenaktensystem wurden definiert. Eine vollständig dezentrale, aber auch eine für den gesamten Wirkungsbereich des Systems zentralisierte Speicherung der Daten wurde ausgeschlossen.

### Problem

Bei der Konzeption der Architektur einer verteilten Krankenakte gibt es sowohl Aspekte die für eine vollständige Zentralisierung bei der Speicherung der Krankenakten-Inhalte sprechen als auch solche, die für eine Beibehaltung der vollständig dezentralisierten fachlich geprägten partiellen Aktensysteme sprechen. Die Einfachheit der Administration spricht für den Aufbau eines zentralen Systems. Aspekte der Datensicherung und Datenübetragungskapazität sprechen häufig für die dezentrale Speicherung, da so die Daten dort gespeichert werden, wo sie erstellt wurden und erwartungsgemäß am häufigsten gebraucht werden. Außerdem ist es im Falle eines Systemdefekts bei dezentraler Datenspeicherung unwahrscheinlich, dass alle Daten zu einem Patiente verloren gehen.

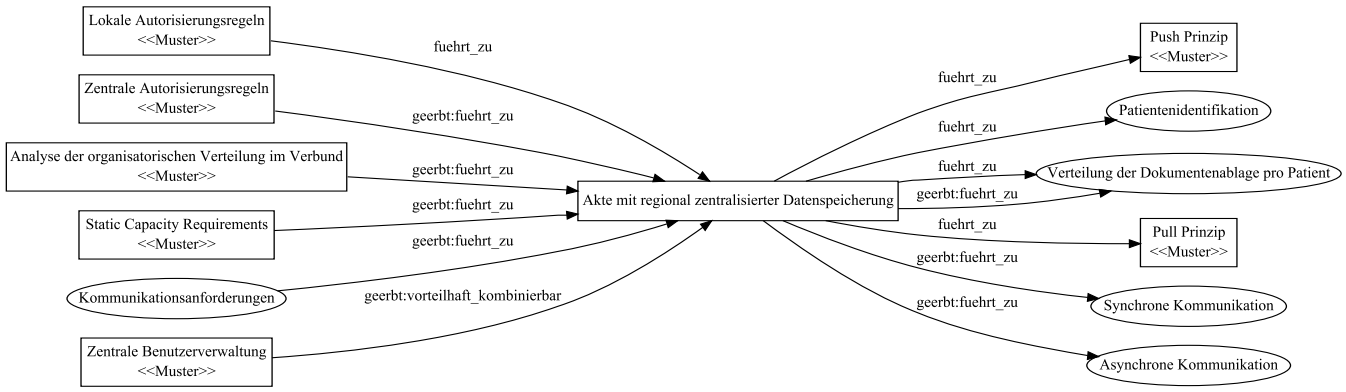
### Lösung

Ist die Auswahl zwischen zentralisierter- und dezentraler Datenspeicherung nicht, oder nur schwer möglich, da aufgrund der pro- und contra Argumente keine eindeutige Entscheidung getroffen werden kann, so sollte als dritte Alternative die regional zentralisierte Datenspeicherung berücksichtigt werden. Dabei lassen sich einige der Vorteile zentralisierter- und dezentraler Datenspeicherung gemeinsam erreichen.

### Beispiel

Ein Beispiel für eine regional zentralisierte Datenspeicherung ist, wenn der Nutzungsbereich einer verteilten Krankenakte (z. B. die Bundesrepublik Deutschland) in Regionen (z. B. Regierungsbezirke) aufgeteilt wird. Für jede dieser Regionen wird ein zentrales Aktensystem installiert. Jeder der Gesundheitsdienstleister nutzt dieses System direkt oder verwendet Systeme, die das System direkt mit den Befunddaten befüllen bzw. diese von dort abfragen. Die Systeme der Regierungsbezirke sind technisch gleich und kommunizieren miteinander, so dass sich die Nutzung des Systems für den Benutzer darstellt, als würde er ein System mit zentralisierter Datenspeicherung verwenden.

### Beziehungen - Grafik



Beziehungen - Detail

- **Lokale Autorisierungsregeln -> Akte mit regional zentralisierter Datenspeicherung** : kombinierbar
  Sollen lokale Autorisierungsregeln verwendet werden, so sind diese mit Akten mit regional zentralisierter Datenspeicherung kombinierbar. Es gilt allerdings die Einschränkung, dass sich in diesem Fall der Begriff lokal ausschließlich auf die Daten verwaltenden Knoten bezieht. Diese sind bei einer Akte mit regional zentralisierter Datenspeicherung allerdings ausschließlich in den regionalen Zentren verfügbar.
- **Akte mit regional zentralisierter Datenspeicherung -> Push Prinzip** : Synchronisation
  Das Push-Prinzip (umgesetzt Beispielsweise durch Point-to-Point-Messaging oder Publish-Subscribe-Messaging) kann in der Architektur einer Akte mit regional zentralisierter Datenspeicherung dazu genutzt werden um z. B. Stammdaten zwischen den regionalen Speicherknoten zu synchronisieren.
- **Akte mit regional zentralisierter Datenspeicherung -> Patientenidentifikation** : falls Patienten nicht bereits eindeutig identifizierbar sind
- **Akte mit regional zentralisierter Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
  Bei der Verwendung einer Akte mit regional zentralisierter Datenspeicherung muss festgelegt werden, ob die Daten zu einem Patienten in einem, dem Patienten fest zugeordneten regionalen Knoten gespeichert werden oder in pro Patient möglicherweise mehreren Knoten abgelegt werden sollen. Der letztere Fall kann z. B. auftreten, wenn der Ablageort nicht dem Patienten regional zugeordnet, sondern der Einrichtung, die den Patienten behandelt regional zugeordnet wird.
- **Akte mit regional zentralisierter Datenspeicherung -> Pull Prinzip** : Datenzugriff
  Der Datenzugriff auf die Inhalte von Akten mit regional zentralisierter Datenspeicherung kann vorteilhaft mittels des Pull-Prinzips (umgesetzt z. B. durch Remote Procedure Invocation) erfolgen.
- **Zentrale Autorisierungsregeln -> Verteilungsarchitektur der Akte** : kombinierbar mit beliebiger Verteilungsarchitektur
  Geerbte Beziehung: fuehrt_zu_beziehung
  Prinzipiell sind zentrale Autorisierungsregeln beliebig mit Verteilungsarchitekturen der Akte kombinierbar, sofern alle beteiligten Knoten einer Architektur mit nicht zentraler Datenspeicherung geeignet sind einen gemeinsamen Dienst zur Verwaltung der Autorisierungsregeln zu verwenden. Da das bei Umgebungen mit einer Vielzahl bereits bestehender Anwendungen selten der Fall ist, existiert die Kombinierbarkeit vorrangig mit den Ausprägungen "Akte mit regional zentralisierter Datenspeicherung" und "Akte mit zentraler Datenspeicherung".
- **Analyse der organisatorischen Verteilung im Verbund -> Verteilungsarchitektur der Akte** : Einrichtungen und deren Leistungsfähigkeit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Leistungsfähigkeit der beteiligten Einrichtungen beeinflusst direkt die Auswahl der Verteilungsarchitektur der Akte. Sind viele Einrichtungen mit niedriger

finanzieller Leistungsfähigkeit beteiligt, so liegt die Wahl einer zentralen oder regional zentralisierten Architektur näher. Sind viele leistungsstarke Einrichtungen beteiligt, so sind diese auch mit höherer Wahrscheinlichkeit dazu fähig einen der dezentralen Knoten zu betreiben.

- **Static Capacity Requirements -> Verteilungsarchitektur der Akte** : Beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Static Capacity Requirements beschreiben die Menge an Daten bzw. Dokumenten, die die verteilte Krankenakte oder einer ihrer Bestandteile zu speichern in der Lage sein muss. Die Menge dieser Daten kann für die Entscheidung zugunsten einer zentralen oder dezentralen Architektur maßgeblich sein.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.
- **Zentrale Benutzerverwaltung -> Verteilungsarchitektur der Akte** : Abhängig von
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wie sinnvoll eine zentrale Benutzerverwaltung ist und mit welchem Aufwand sie sich realisieren lässt, ist abhängig von der gewählten Verteilungsarchitektur der Akte.
- **Verteilungsarchitektur der Akte -> Verteilung der Dokumentenablage pro Patient** : unterschiedlich kombinierbar
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilungsarchitektur der Akte -> Synchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.

## *8. Akte mit vollständig dezentraler Datenspeicherung*

Englische Bezeichnung

- Health Record with fully decentralized Data Storage

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Verteilungsarchitektur der Akte

Kontext

Die Anforderungen an das zukünftige verteilte Krankenaktensystem wurden definiert. Die Beibehaltung der existierenden Systeme als primäre Datensenken ist in den Anforderungen hoch priorisiert. Es bestehen evtl. bereits Teile einer Kommunikations-Infrastruktur zu Übermittlung bzw. Synchronisation von Daten zwischen den existierenden abteilungs- oder hausinternen Systemen. Eine umfassende Modifikation der bestehenden Strukturen ist entsprechend der ermittelten Rahmenbedingungen nicht möglich.

Problem
Die Rahmenbedingungen erlauben keine Zentralisierung der Aktendaten. Auch eine teilweise Zentralisierung ist nicht möglich.

Lösung
Die Architektur einer Krankenakte mit vollständig dezentraler Datenspeicherung kann auf mindestens zwei verschiedene Weisen aufgebaut werden. Alle zwei Ansätze weisen unterschiedliche Vor- und Nachteile auf und sind dadurch für unterschiedliche Einsatzszenarien geeignet.

Lösungsansatz 1:
Die Stammdaten der dezentralen Datenbestände werden durch asynchron übermittelte Nachrichten synchronisiert. Bei der Anlage von Dokumenten wird bereits festgelegt, wer deren Inhalt erhalten soll. Basierend auf dieser Information werden neue Dokumente durch Nachrichten an die festgelegten Empfänger übermittelt. Vorher nicht festgelegte Empfänger können Dokumente nur erhalten, wenn Sie die Übermittlung des Dokuments bei dessen Ersteller oder einem der Empfänger beantragt und dieser das Dokument weiterleitet.
**Vorteile:**

- Jedes System ist eigenständig funktionsfähig und hält alle Daten, die es zum Betrieb benötigt selbst vor.
- Alle Versorgungseinheiten bleiben auch bei einem Ausfall der Infrastruktur, die die Daten synchronisiert, vollständig handlungsfähig.
- Es ist keine gemeinsame Verwaltung von Benutzern und Rechten erforderlich.

**Nachteile:**

- Die Synchronisation der Daten ist mit steigender Zahl der Knoten zunehmend fehleranfällig.
- Die Synchronisation der Daten generiert mit steigender Zahl der Knoten mehr Last auf den einzelnen Systemen.
- Bereits versandte Dokumente müssen im Falle einer Erweiterung oder Änderung zu Zwecken der Synchronisation nochmal versendet werden. Daraus resultiert, dass die

Dokumente in allen Empfängerknoten versioniert werden müssen um durch erneute Übermittlung eingebrachte Änderungen nachvollziehbar zu halten.
- Die Übertragung eines Dokuments an einen Empfänger kann durch den Patienten nicht widerrufen werden.

Lösungsansatz 2:
Alle Dokumente werden ausschließlich auf dem System dauerhaft gespeichert, auf dem sie erstellt wurden. Systeme von Benutzern, die keinen direkten Zugriff auf ein bestimmtes Dokument haben greifen, abhängig von den Berechtigungen des Benutzers, indirekt über die Verwendung von Mitteln synchroner Kommunikation auf die Daten zu. Durch die Verwendung z. B. des Adapter-Registry-Patterns lässt sich auch in dieser hoch verteilten Architektur eine gemeinsame Zugriffsschnittstelle und daraus resultierend, ortstransparenter Zugriff ermöglichen.
**Vorteile:**

- Redundanzfreie Ablage der Dokumente, daraus resultierend
  - keine Probleme mit der Synchronisation von Änderungen
  - Dokumente müssen nur in einem System versioniert werden
- Datenübertragung erfolgt nur bei Bedarf
- Zugriffsberechtigungen können bei jedem Zugriff geprüft werden

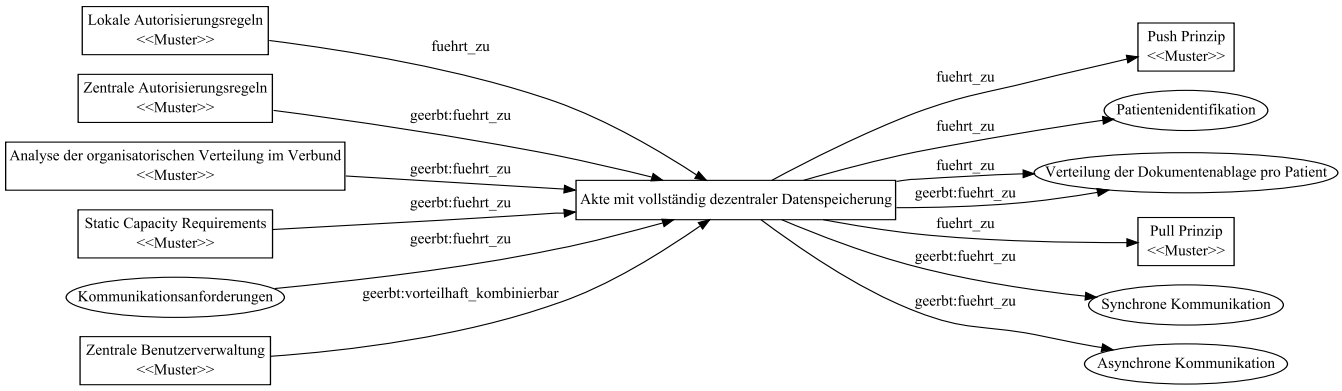
**Nachteile:**

- Bei einem Ausfall der Kommunikationsinfrastruktur kann ggf. auf bereits verwendete fremde Dokumente nicht zugegriffen werden.
- Das Suchen nach Dokumenten kann, wenn es ohne einen zentralen Index ausgeführt wird, hohe Last auf den beteiligten Knoten verursachen.
- Benutzer müssen über alle Knoten hinweg eindeutig identifizierbar sein, um Berechtigungen auswerten zu können.

In beiden Ansätzen besteht grundsätzlich das Problem der eindeutigen Identifikation von Patienten.

Beispiel
Ein Beispiel für den Lösungsansatz 2 wird in Kapitel 6 vorgestellt.

Beziehungen - Grafik



Beziehungen - Detail

- **Lokale Autorisierungsregeln -> Akte mit vollständig dezentraler Datenspeicherung** : kombinierbar
  Lokale Autorisierungsregeln können in einer Akte mit vollständig dezentraler Datenspeicherung umgesetzt werden. Dann besitzt jeder Knoten seine eigenen Zugriffsregeln, die den Zugriff auf die in ihm enthaltenen Objekte (med. Dokumente) regeln.
- **Akte mit vollständig dezentraler Datenspeicherung -> Push Prinzip** : Ansatz 1: Synchronisation
  Bei verteilten Krankenakten mit vollständig dezentraler Datenspeicherung die den Ansatz 1 des Patterns oder eine abgewandelte Form davon verwenden, folgt die Kommunikation dem Push Prinzip. Die Architektur der Akte erfordert eine vollständige Synchronisation der Stammdaten zwischen allen Knoten, so dass Änderungen immer an alle Knoten propagiert werden müssen. Auch Dokumente werden nach dem Push-Prinzip an alle als berechtigt identifizierten Empfänger versendet.
- **Akte mit vollständig dezentraler Datenspeicherung -> Patientenidentifikation** : falls Patienten nicht bereits eindeutig identifizierbar sind
- **Akte mit vollständig dezentraler Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
- **Akte mit vollständig dezentraler Datenspeicherung -> Pull Prinzip** : Ansatz 2: Zugriff auf Inhalte
  Wird der Ansatz 2 einer Akte mit vollständig dezentraler Datenspeicherung realisiert, so folgt sämtliche Kommunikation dem Pull-Prinzip. Berechtigte Benutzer greifen durch gezielte Anfragen von entfernten Systemen auf enthaltene Dokumente zu. Dies entspricht aktivem Abholen bei Bedarf (Pull).
- **Zentrale Autorisierungsregeln -> Verteilungsarchitektur der Akte** : kombinierbar mit beliebiger Verteilungsarchitektur
  Geerbte Beziehung: fuehrt_zu_beziehung
  Prinzipiell sind zentrale Autorisierungsregeln beliebig mit Verteilungsarchitekturen der Akte kombinierbar, sofern alle beteiligten Knoten einer Architektur mit nicht zentraler Datenspeicherung geeignet sind einen gemeinsamen Dienst zur Verwaltung der Autorisierungsregeln zu verwenden. Da das bei Umgebungen mit einer Vielzahl bereits bestehender Anwendungen selten der Fall ist, existiert die Kombinierbarkeit vorrangig mit den Ausprägungen "Akte mit regional zentralisierter Datenspeicherung" und "Akte mit zentraler Datenspeicherung".
- **Analyse der organisatorischen Verteilung im Verbund -> Verteilungsarchitektur der Akte** : Einrichtungen und deren Leistungsfähigkeit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Leistungsfähigkeit der beteiligten Einrichtungen beeinflusst direkt die Auswahl der Verteilungsarchitektur der Akte. Sind viele Einrichtungen mit niedriger finanzieller Leistungsfähigkeit beteiligt, so liegt die Wahl einer zentralen oder regional zentralisierten Architektur näher. Sind viele leistungsstarke Einrichtungen

beteiligt, so sind diese auch mit höherer Wahrscheinlichkeit dazu fähig einen der dezentralen Knoten zu betreiben.

- **Static Capacity Requirements -> Verteilungsarchitektur der Akte** : Beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Static Capacity Requirements beschreiben die Menge an Daten bzw. Dokumenten, die die verteilte Krankenakte oder einer ihrer Bestandteile zu speichern in der Lage sein muss. Die Menge dieser Daten kann für die Entscheidung zugunsten einer zentralen oder dezentralen Architektur maßgeblich sein.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.
- **Zentrale Benutzerverwaltung -> Verteilungsarchitektur der Akte** : Abhängig von
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wie sinnvoll eine zentrale Benutzerverwaltung ist und mit welchem Aufwand sie sich realisieren lässt, ist abhängig von der gewählten Verteilungsarchitektur der Akte.
- **Verteilungsarchitektur der Akte -> Verteilung der Dokumentenablage pro Patient** : unterschiedlich kombinierbar
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilungsarchitektur der Akte -> Synchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.

## *9. Akte mit zentraler Datenspeicherung*

Englische Bezeichnung

- Health Record with centralized Data Storage

Schwerpunkt(e)

- Kommunikation

### Gruppe(n)

- Verteilungsarchitektur der Akte

### Kontext

Die Anforderungen an das zukünftige verteilte Krankenaktensystem wurden definiert. Eine vollständig dezentrale und eine regional zentralisierte Datenspeicherung wurden ausgeschlossen.

### Problem

Die Architektur für eine verteilte Krankenakte muss so umgesetzt werden, dass die Datenspeicherung für alle beteiligten Gesundheitsdienstleister an einem zentralen Ort erfolgt.

### Lösung

Grundsätzlich hat eine Architektur mit zentraler Datenspeicherung den Vorteil, dass alle Daten gemeinsam in einer Datensenke verfügbar sind. Dadurch ist es möglich zentrale Backup-Verfahren, Sicherheitsmechanismen usw. umzusetzen. Die Anzahl der unterschiedlichen, zu betreibenden Verfahren wird dadurch geringer, was wiederum die Risiken von Fehlern durch fehlerhafte Administration reduziert. Die Architektur mit zentraler Datenspeicherung weist den Nachteil auf, dass die Nutzung des Systems stärker als bei dezentralen Ansätzen von der Verfügbarkeit der Netzwerkinfrastruktur abhängig ist. Die Umsetzung einer Akte mit zentraler Datenhaltung kann in unterschiedlichen Formen erfolgen.

### Lösungsansatz 1:

Lokale Systeme wie Krankenhausinformationssysteme bzw. Abteilungsinformationssysteme liefern die Daten nach deren Erstellung an den zentralen Datenspeicher.

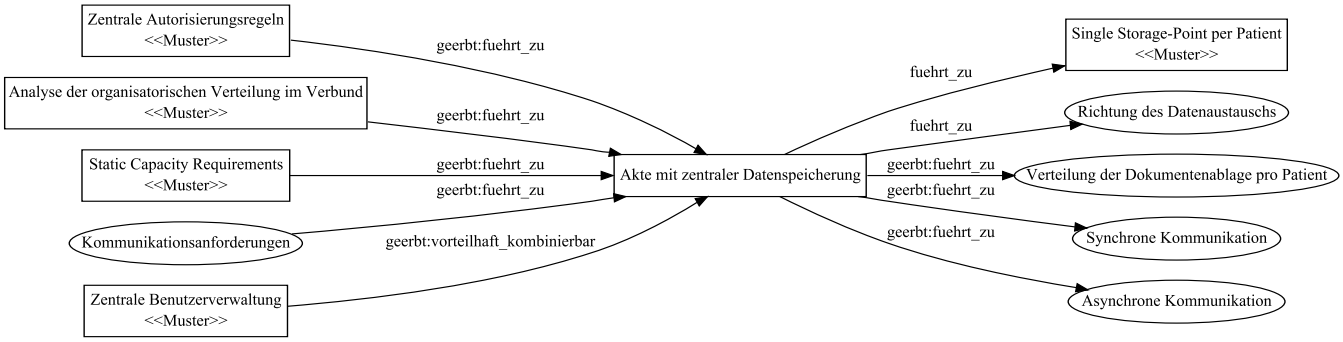
- Daten werden nach dem Push-Prinzip (mittels Point-to-Point Messaging) aus den lokalen Systemen an die zentrale Datensenke geliefert.
- Der Zugriff auf die Daten erfolgt nach dem Pull-Prinzip (mittels Remote Procedure Invocation, bei großen Dokumenten ggf. asynchrones Request/Reply-Messaging)

### Lösungsansatz 2:

Der zentrale Datenspeicher ist die einzige Datensenke. Die Benutzer in den verschiedenen beteiligten Institutionen greifen ausschließlich mittels Client-Software oder mittels zwischenspeichernden Proxy-Servern und Client-Software auf die Daten des zentralen Knoten zu.

- Die Daten werden mittels entsprechender Client-Software direkt in der zentralen Datensenke erzeugt.
- Die Kommunikation zwischen Client und zentralem Datenspeicher kann durch Remote Procedure Invocation oder asynchrone Request Reply Protokolle erfolgen
- Der Zugriff auf die Daten erfolgt nach dem Pull-Prinzip.

Beziehungen - Grafik




Beziehungen - Detail

- **Akte mit zentraler Datenspeicherung -> Single Storage-Point per Patient** : kann nur als realisiert werden
  Wenn nur ein einziger (zentraler) Ort zur Datenspeicherung im gesamten Aktensystem existiert, können die Daten zu einem Patienten auch nur an diesem einen Ort gespeichert sein.
- **Akte mit zentraler Datenspeicherung -> Richtung des Datenaustauschs** : Situationsabhängige Auswahl
  Bei einer Akte mit zentraler Datenspeicherung können die beiden Prinzipien Push und Pull einander ergänzend verwendet werden. Je nach gewähltem Lösungsansatz wird die Kombination beider Prinzipien etwas unterschiedlich gestaltet. Grundsätzlich werden Daten nach der Erstellung an den zentralen Speicherort übertragen oder durch einzelne Übertragungen von Teilen Stückweise dort erstellt. Der lesende bzw. suchende Zugriff auf die Daten erfolgt in beiden Fällen via Pull.
- **Zentrale Autorisierungsregeln -> Verteilungsarchitektur der Akte** : kombinierbar mit beliebiger Verteilungsarchitektur
  Geerbte Beziehung: fuehrt_zu_beziehung
  Prinzipiell sind zentrale Autorisierungsregeln beliebig mit Verteilungsarchitekturen der Akte kombinierbar, sofern alle beteiligten Knoten einer Architektur mit nicht zentraler Datenspeicherung geeignet sind einen gemeinsamen Dienst zur Verwaltung der Autorisierungsregeln zu verwenden. Da das bei Umgebungen mit einer Vielzahl bereits bestehender Anwendungen selten der Fall ist, existiert die Kombinierbarkeit vorrangig mit den Ausprägungen "Akte mit regional zentralisierter Datenspeicherung" und "Akte mit zentraler Datenspeicherung".
- **Analyse der organisatorischen Verteilung im Verbund -> Verteilungsarchitektur der Akte** : Einrichtungen und deren Leistungsfähigkeit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Leistungsfähigkeit der beteiligten Einrichtungen beeinflusst direkt die Auswahl der Verteilungsarchitektur der Akte. Sind viele Einrichtungen mit niedriger finanzieller Leistungsfähigkeit beteiligt, so liegt die Wahl einer zentralen oder regional zentralisierten Architektur näher. Sind viele leistungsstarke Einrichtungen beteiligt, so sind diese auch mit höherer Wahrscheinlichkeit dazu fähig einen der dezentralen Knoten zu betreiben.

- **Static Capacity Requirements -> Verteilungsarchitektur der Akte** : Beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Static Capacity Requirements beschreiben die Menge an Daten bzw. Dokumenten, die die verteilte Krankenakte oder einer ihrer Bestandteile zu speichern in der Lage sein muss. Die Menge dieser Daten kann für die Entscheidung zugunsten einer zentralen oder dezentralen Architektur maßgeblich sein.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.
- **Zentrale Benutzerverwaltung -> Verteilungsarchitektur der Akte** : Abhängig von
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wie sinnvoll eine zentrale Benutzerverwaltung ist und mit welchem Aufwand sie sich realisieren lässt, ist abhängig von der gewählten Verteilungsarchitektur der Akte.
- **Verteilungsarchitektur der Akte -> Verteilung der Dokumentenablage pro Patient** : unterschiedlich kombinierbar
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilungsarchitektur der Akte -> Synchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.

## *10. Analyse der betroffenen Behandlungspfade*

Englische Bezeichnung

- Analysis of relevant Clinical Pathways

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Analyse flexibilitätsbeeinflussender Faktoren

### Kontext

Behandlungspfade sind medizinische Geschäftsprozesse. Sie bilden komplexe Behandlungsprozesse ab. Im Falle von verteilten Krankenakten kann i. d. R. von einrichtungsübergreifenden Behandlungspfaden (siehe Böckmann et al. (Böckmann & Houta 2008; Eckenbach & Böckmann 2008)) gesprochen werden. Die Analyse dieser übergreifenden und/oder der lokalen Behandlungspfade führt zu Erkenntnissen über die mittels der verteilten Akte abzubildenden Prozesse.
Im Rahmen dieses Erkenntnisprozesses wird auf eine existierende Analyse der organisatorischen Verteilung im Verbund zurückgegriffen. Die Ergebnisse werden gemeinsam mit denen der betroffenen Subsysteme usw. zur Ermittlung der Flexibilitätsanforderungen verwendet.

### Problem

Wie können die für die Entwicklung der verteilten Krankenakte relevanten Behandlungspfade so erfasst und dokumentiert werden, dass sich ein Hilfsmittel für die Konzeption der Software-Architektur der verteilten Krankenakte ergibt.

### Lösung

Die Lösung zur Analyse der betroffenen Behandlungspfade ist als ein, in mehrere aufeinander aufbauende Schritte gegliedertes Konzept realisiert:

### Vorbedingungen

- Im Rahmen der Analyse der Organisatorischen Verteilung im Verbund wurden die an der verteilten Krankenakte beteiligten Organisationen und Organisationseinheiten identifiziert.
- Der Zweck den die verteilte Krankenakte nach ihrer Fertigstellung erfüllen soll ist bekannt.
- Es ist bekannt, ob es sich um eine vollständige oder partielle Akte handeln soll.

### Schritt 1

Ermittlung aller Leistungen, die durch die verteilte Krankenakte unterstützt werden sollen. Die Ermittlung wird pro beteiligte Organisationseinheit durchgeführt. Das Ergebnis ist pro Organisationseinheit eine Liste mit betroffenen Aufgaben.

### Schritt 2

Ermittlung aller in den betroffenen Organisationseinheiten bereits dokumentierten Behandlungspfade.
Hinweis: Behandlungspfade können in zwei Formen existieren. Als bereits dokumentierte Behandlungspfade und als nicht formalisierte, intuitiv befolgte Behandlungspfade. Hier sind ausschließlich bereits dokumentierte Pfade gesucht.

### Schritt 3

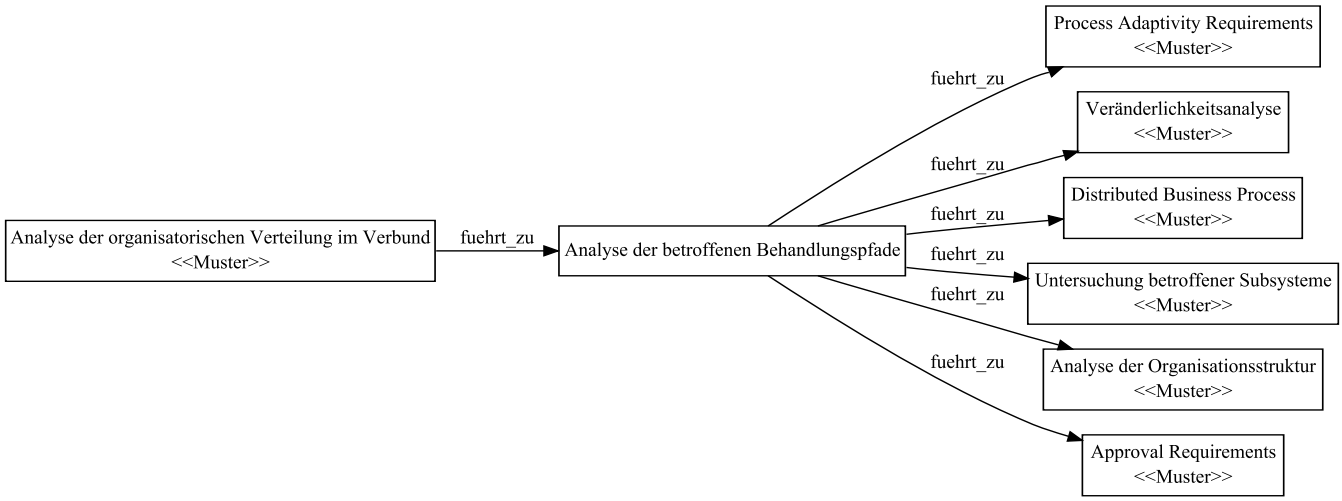
Abgleich zwischen ermittelten Leistungen und Pfaden. Das Ergebnis des Abgleichs ist eine Liste mit Arbeitsschritten, die noch nicht durch dokumentierte Pfade beschrieben ist.

Schritt 4

Abbildung der zukünftig durch die verteilte Krankenakte gewünschten Behandlungspfade auf Basis der bestehenden Behandlungspfad-Dokumentation und der noch nicht in bestehenden Behandlungspfaden beschriebenen Aktionen. Die Dokumentation sollte sowohl, den Ablauf visualisierende Diagramme (z. B. Aktivitätsdiagramme oder ereignisgesteuerte Prozessketten (EPK)) als auch eine erläuternde textuelle Dokumentation mit Bezugnahme auf die Diagramme enthalten.

Beispiel

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der organisatorischen Verteilung im Verbund -> Analyse der betroffenen Behandlungspfade** : Input: Beteiligte Einrichtungen mit Behandlungspfaden
  Die Analyse der organisatorischen Verteilung im Verbund liefert die Menge der an der verteilten Krankenakte beteiligten Einrichtungen, deren betroffene Behandlungspfade analysiert werden sollen. Außerdem wird zu jeder Einrichtung bereits deren Leistungsspektrum eingegrenzt. Das hilft bei der Findung relevanter Behandlungspfade.
- **Analyse der betroffenen Behandlungspfade -> Process Adaptivity Requirements** : Input: Pfade und letzte Änderungen an Pfaden
  Die Analyse der betroffenen Behandlungspfade liefert die relevanten Behandlungspfade (medizinische Prozesse) für die Process Adaptivity Requirements festgelegt werden müssen.
- **Analyse der betroffenen Behandlungspfade -> Veränderlichkeitsanalyse** : Input: Behandlungspfade
  Behandlungspfade können, vor allem wenn sie einrichtungsübergreifend definiert sind, häufigen Änderungen unterworfen sein. Typische Änderungsgründe für einrichtungsübergreifende Behandlungspfade sind die Veränderung der vertraglichen Verbindungen zwischen den Einrichtungen oder die Einführung neuer Behandlungsmethoden. Das Pattern "Analyse veränderlicher Bestandteile" dient dazu

zu ermitteln, welche Bestandteile der Anforderungen häufigen Veränderungen unterworfen sind und welche als relativ konstant betrachtet werden können.

- **Analyse der betroffenen Behandlungspfade -> Distributed Business Process** : Wenn über mehrere Subsysteme verteilt
  Wenn die Analyse der betroffenen Behandlungspfade ergeben hat, dass sich die Prozesse über mehrere beteiligte Subsysteme erstrecken, kann im Verlauf der Architekturkonzeption die Anwendung des Distributed-Business-Process-Patterns vorteilhaft sein.
- **Analyse der betroffenen Behandlungspfade -> Untersuchung betroffener Subsysteme** : Input: KA-relevante Aufgaben
  Die Analyse der betroffenen Behandlungspfade liefert für die Untersuchung der betroffenen Subsysteme die Menge der für die Entwicklung der verteilten Krankenakte relevanten Aufgaben. Relevant sind dabei alle Aufgaben, die in den Behandlungspfaden, die durch die verteilte Krankenakte unterstützt werden sollen, enthalten sind.
- **Analyse der betroffenen Behandlungspfade -> Analyse der Organisationsstruktur** : Ermittlung von Akteuren und Aufgabengebieten
  Die Analyse der Organisationsstruktur verfeinert die Analyse der betroffenen Behnadlungspfade durch hinzufügen von Akteuren und Aufgabengebieten zu den durch die Analyse der betroffenen Behandlungspfade ermittelten Pfaden und Aufgaben.
- **Analyse der betroffenen Behandlungspfade -> Approval Requirements** : Fachliche Abläufe
  Die analysierten Behandlungspfade enthalten ggf. Aktivitäten in denen ein systemseitig unterstütztes Bestägigungsverfahren notwendig oder hilfreich ist. Solche Stellen führen zur Definition von Approval Requirements.

## *11. Analyse der Organisationsstruktur*

Englische Bezeichnung

- Analysis of Organizational Structure

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Analyse der Sicherheitssituation

Kontext

Aus der Analyse der organisatorischen Verteilung im Verbund sind alle beteiligten Einrichtungen bekannt. Die von der Einführung bzw. Entwicklung einer verteilten

Krankenakte betroffenen Behandlungspfade wurden analysiert. Die Sicherheitsrelevanten Assets wurden bestimmt und bewertet.

### Problem

Für die Planung sicherheitsrelevanter Architekturbestandteile fehlen noch Informationen zu den beteiligten Akteuren, deren Aufgaben (genauer, Klassen von Aufgaben) sowie eine Zuordnung zwischen Akteuren und deren Aufgaben. Außerdem ist die organisatorische Zuordnung der Akteure noch nicht beschrieben.

### Lösung

#### Schritt 1

Aus der Analyse der organisatorischen Verteilung im Verbund sind alle beteiligten Einrichtungen mit ihrem Behandlungsspektrum bekannt. Für jede der bekannten Einrichtungen wird deren Abteilungsstruktur erfasst.

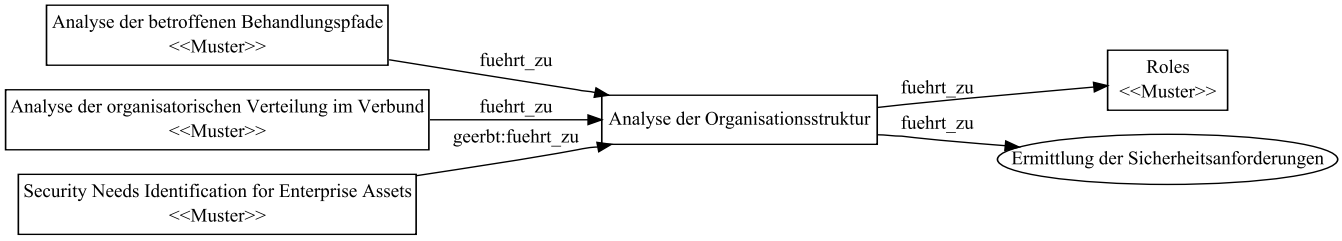
#### Schritt 2

Basierend auf den Ergebnissen der Untersuchung der relevanten Behandlungspfade wird eine Liste der Akteure aufgebaut. Ein Akteur ist dabei definiert durch seine organisatorische Zuordnung (Einrichtung + Abteilung + ggf. Funktion) und seine Qualifikation. Ein Beispiel für einen solchen Akteur ist "Arzt der Notaufnahme im Krankenhaus Musterstadt".

#### Schritt 3

Jedem Akteur aus Schritt 2 werden seine konkreten Aufgaben im Rahmen der relevanten Behandlungspfade zugeordnet. Das Ergebnis ist eine Baumstruktur aus Akteuren mit deren zugeordneten Aufgaben.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Analyse der betroffenen Behandlungspfade -> Analyse der Organisationsstruktur** : Ermittlung von Akteuren und Aufgabengebieten
  Die Analyse der Organisationsstruktur verfeinert die Analyse der betroffenen Behnadlungspfade durch hinzufügen von Akteuren und Aufgabengebieten zu den durch die Analyse der betroffenen Behandlungspfade ermittelten Pfaden und Aufgaben.
- **Analyse der organisatorischen Verteilung im Verbund -> Analyse der Organisationsstruktur** : Grobanalyse zu Detailanalyse
  Aus der Analyse der organisatorischen Verteilung im Verbund sind alle beteiligten Einrichtungen mit ihrem Behandlungsspektrum bekannt. In der Analyse der

Organisationsstruktur werden aufbauend auf diesen Informationen, gemeinsam mit denen aus der Analyse der betroffenen Behandlungspfade Akteure ermittelt und eine Zuordnung zwischen Aufgaben und Akteuren erstellt. Diese Zuordnung bietet z. B. dem Roles-Pattern die Basis für die Spezifikation konkreter Sicherheitsrollen.

- **Analyse der Organisationsstruktur -> Roles** : Input: Akteure und Aufgabengebiete
  Das Analyse-der-Organisationsstruktur-Pattern liefert durch seine Anwendendung eine Auflistung von Akteuren und deren Aufgabengebiete. Aufgabe des Roles-Patterns ist es, diese zu abstrahieren um sie zu allgemeinen, für die Anforderungsdefinition standardisierten Rollen zusammenzufassen.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Security Needs Identification for Enterprise Assets -> Analyse der Sicherheitssituation** : Grobanalyse zu Detailanalyse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Security-Needs-Identification-for-Enterprise-Assets-Pattern dient der Identifikation und Bewertung gefährdeter Güter, Anlagen, Personen und Daten in den untersuchten medizinischen Versorgungseinrichtungen. Die ermittelten Gefährdeten, sowie das ebenfalls ermittelte Einflusspotenzial auf den Gesamtbetrieb dienen dazu im Rahmen der Analyse der Sicherheitssituation ermittelte Sicherheitsrisiken und andere Einflussfaktoren nach ihrer Wichtigkeit zu bewerten.

## *12. Analyse der organisatorischen Verteilung im Verbund*

Englische Bezeichnung

- Analysis of the Organizational Responsibility Assignment within a Union of Healthcare Providers

Schwerpunkt(e)

- Kommunikation
- Zuverlässigkeit
- Flexibilität

Gruppe(n)

- Analyse flexibilitätsbeeinflussender Faktoren
- Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl.
- Analyse des Kommunikationsbedarfs

### Kontext

Einer verteilten Krankenakte liegt üblicherweise ein Verbund von Einrichtungen oder Organisationseinheiten einer Einrichtung zugrunde. Die Planung zur Entwicklung einer geeigneten IT-Lösung zur Umsetzung der verteilten Krankenakte kann mit der Anwendung dieses Musters begonnen werden.

### Problem

Zu Beginn der Planung einer verteilten Krankenakte ist über die Details dieses Verbunds (siehe Kontext) wenig bekannt. Der Aufbau des Verbunds ist eine wichtige Information für die Analyse der Rahmenbedingungen, die Definition der Anforderungen, aber auch der Gestaltung der Architektur der Anwendung.

### Lösung

#### Schritt 1

Alle Einrichtungen, die an der verteilten Krankenakte beteiligt sein sollen, werden in einer Liste erfasst.

#### Schritt 2:

Jeder Einrichtung in der Liste wird ihr Leistungsspektrum hinzugefügt. Die Angabe des Leistungsspektrums muss in dieser Phase der Analyse nicht zu detailliert sein. Die Verwendung von Oberbegriffen hält die Aufzählung besonders bei großen Einrichtungen übersichtlicher.

#### Schritt 3

Die Einrichtungen werden in Größenkategorien nach ihrer finanziellen Leistungsfähigkeit (für den Anteil, den die Einrichtung an der Umsetzung der verteilten Akte leisten kann) und ihrem Leistungsumfang (für den Anteil an der Nutzung und dem Analyseaufwand) klassifiziert. Für den Bedarf in der weiteren Analyse dürften jeweils 3-4 Größenkategorien ausreichend sein.

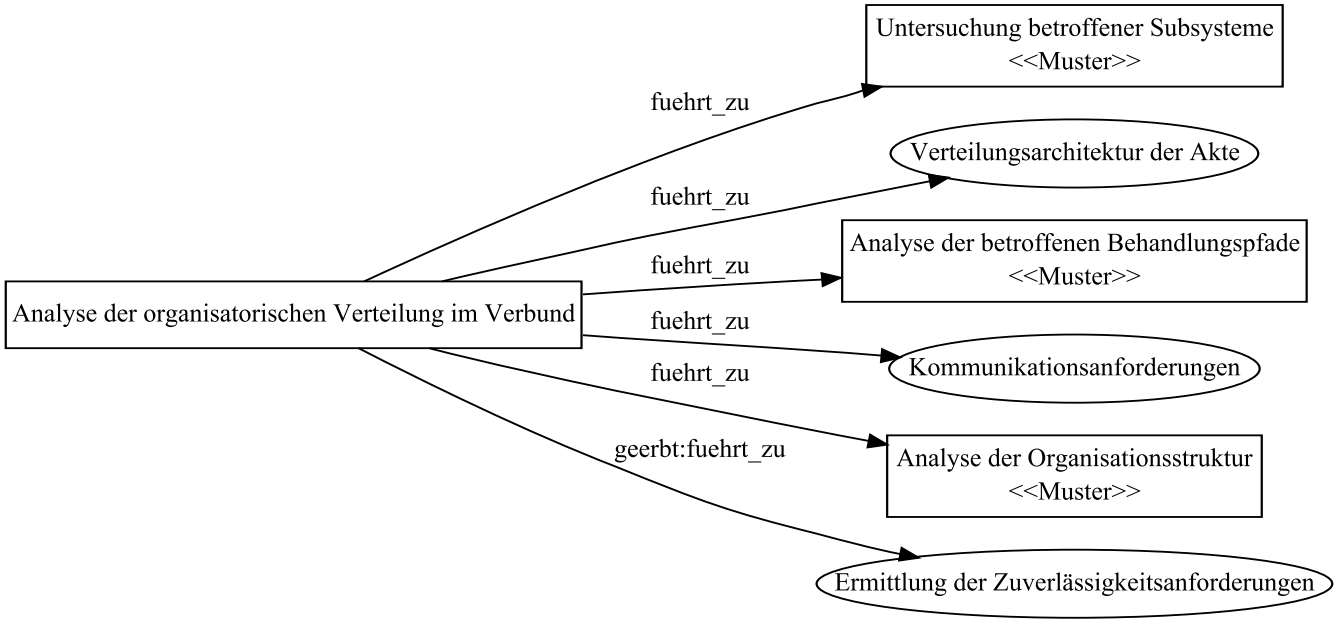
Das Ergebnis der Anwendung dieses Patterns sind drei Listen:

- Einrichtungen nach Größe
- Einrichtungen nach Leistungsumfang
- Einrichtungen mit Leistungen (in Überbegriffen)

Mit dem Ergebnis sollte z. B. bei der Analyse der Behandlungspfade schnell ermittelt werden können, welche Einrichtungen für welchen Pfad relevant sind und welche aus der jeweiligen Untersuchung von vorneherein ausgeklammert werden können.

Beispiel

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der organisatorischen Verteilung im Verbund -> Untersuchung betroffener Subsysteme** : Input: Einrichtungen mit Subsystemen
  Im Rahmen der Analyse der organisatorischen Verteilung werden an der verteilten Krankenakte beteiligte Einrichtungen ermittelt. Die IT-Systeme dieser Einrichtungen werden im Rahmen der Untersuchung betroffener Subsysteme untersucht.
- **Analyse der organisatorischen Verteilung im Verbund -> Verteilungsarchitektur der Akte** : Einrichtungen und deren Leistungsfähigkeit
  Die Leistungsfähigkeit der beteiligten Einrichtungen beeinflusst direkt die Auswahl der Verteilungsarchitektur der Akte. Sind viele Einrichtungen mit niedriger finanzieller Leistungsfähigkeit beteiligt, so liegt die Wahl einer zentralen oder regional zentralisierten Architektur näher. Sind viele leistungsstarke Einrichtungen beteiligt, so sind diese auch mit höherer Wahrscheinlichkeit dazu fähig einen der dezentralen Knoten zu betreiben.
- **Analyse der organisatorischen Verteilung im Verbund -> Analyse der betroffenen Behandlungspfade** : Input: Beteiligte Einrichtungen mit Behandlungspfaden
  Die Analyse der organisatorischen Verteilung im Verbund liefert die Menge der an der verteilten Krankenakte beteiligten Einrichtungen, deren betroffene Behandlungspfade analysiert werden sollen. Außerdem wird zu jeder Einrichtung bereits deren Leistungsspektrum eingegrenzt. Das hilft bei der Findung relevanter Behandlungspfade.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung

im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.

- **Analyse der organisatorischen Verteilung im Verbund -> Analyse der Organisationsstruktur** : Grobanalyse zu Detailanalyse
  Aus der Analyse der organisatorischen Verteilung im Verbund sind alle beteiligten Einrichtungen mit ihrem Behandlungsspektrum bekannt. In der Analyse der Organisationsstruktur werden aufbauend auf diesen Informationen gemeinsam mit denen aus der Analyse der betroffenen Behandlungspfade Akteure ermittelt und eine Zuordnung zwischen Aufgaben und Akteuren erstellt. Diese Zuordnung bietet z. B. dem Roles-Pattern die Basis für die Spezifikation konkreter Sicherheitsrollen.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.

## *13. Approval Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 318

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Zuverlässigkeit

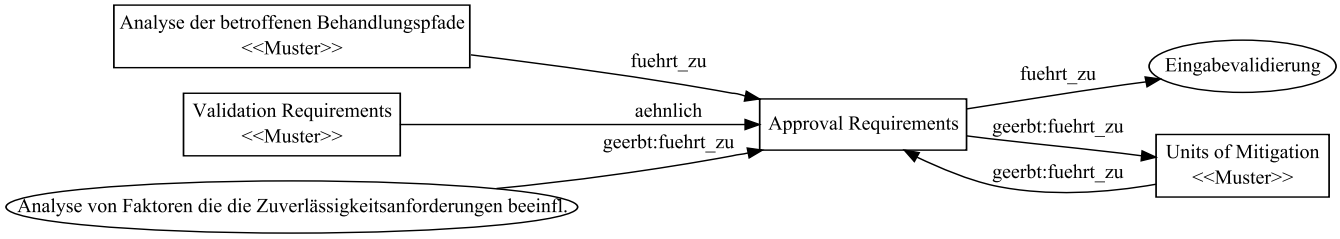
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Approval Requirements" erläutert die Spezifikation von Anforderungen, die definieren, wann in einer Software Aktionen von Benutzern oder eingegebene Daten durch eine dritte Person überprüft werden müssen, bevor sie tatsächlich im System wirksam werden dürfen.

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der betroffenen Behandlungspfade -> Approval Requirements** :
  Fachliche Abläufe
  Die analysierten Behandlungspfade enthalten ggf. Aktivitäten in denen ein systemseitig unterstütztes Bestägigungsverfahren notwendig oder hilfreich ist. Solche Stellen führen zur Definition von Approval Requirements.
- **Validation Requirements -> Approval Requirements** :
  Approval Requirements beschreiben die Anforderungen an eine Software bezüglich der Prüfung eingegebener Daten durch Benutzer, z. B. im Rahmen eines Approval Workflow, während Validation Requirements die Anforderungen an eine automatische Überprüfung (Validierung) eingegebener Daten zum Zeitpunkt der Eingabe mittels Regeln beschreiben.
- **Approval Requirements -> Eingabevalidierung** : Anforderungen für spezielle Eingabevalidierung
  Die Nutzung eines Approval Workflow, also der Bestätigung von eingegebenen Werten durch einen weiteren Benutzer (i.d.R. einen fachlichen Experten), ist eine spezielle Form der Eingabevalidierung.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.

- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input:
  Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## *14. Approval Workflow*

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

- Eingabevalidierung

Kontext

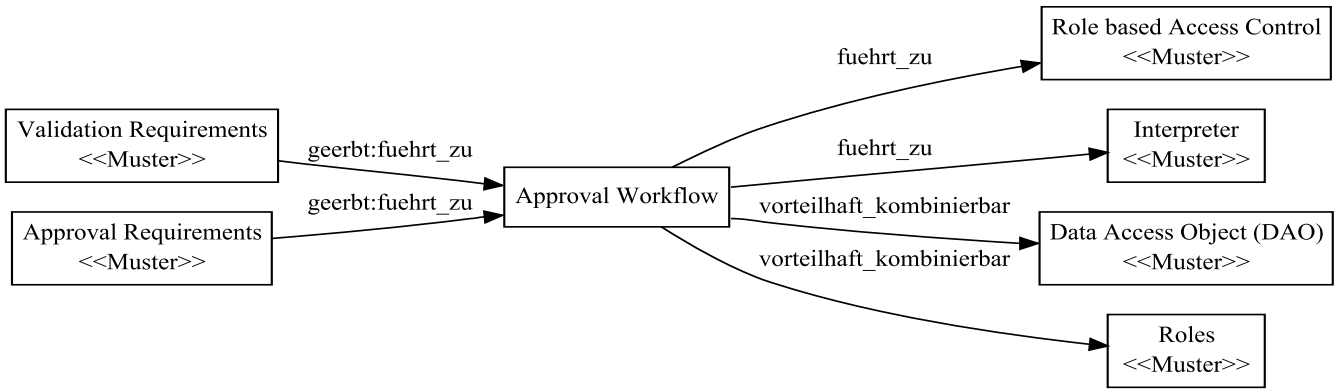
Mindestens ein Anwendungsfall im System benötigt nach der Eingabe von Daten deren Bestätigung durch einen zusätzlichen Benutzer, bevor die Daten ihrer Verwendung zugeführt werden dürfen. Die daraus resultierenden Approval Requirements wurden entsprechend des gleichnamigen Patterns spezifiziert.

Problem
Die Überprüfung einer Eingabe durch einen Dritten ist keine Aufgabe, die ausschließlich systemintern erfolgen kann. Das System muss einen oder mehrere geeignete Benutzer auswählen und diese aktiv über ihre Aufgabe, das Bestätigen eben eingegebener Daten informieren. Außerdem können die Regeln, welche Benutzer zur Bestätigung der Daten heranzuziehen sind, sich sowohl abhängig vom Inhalt der Daten als auch von regelmäßigen organisatorischen Veränderungen ändern.

Lösung
Durch die verschiedenen Veränderlichkeiten, denen der Überprüfungsvorgang ausgesetzt ist, ist es sinnvoll den Vorgang konfigurierbar, also durch Regeln beschreibbar zu gestalten. Die Regeln werden nicht direkt im Quellcode der Anwendung umgesetzt. Die Anwendung selbst enthält stattdessen einen Interpreter, der die entsprechenden Regeln zur Laufzeit interpretiert und die Daten entsprechend dem Ergebnis einem passenden Benutzer zur Überprüfung vorlegt. Um auch Vertretungsfälle und organisatorische Veränderungen zu berücksichtigen, ist es vorteilhaft, die Regeln auf Basis von Rollen und nicht von Benutzern zu formulieren.

Beispiel

Beziehungen - Grafik




Beziehungen - Detail

- **Approval Workflow -> Role based Access Control** : Rollenbasierte Zuordnung des Prüfenden
  Zuständigkeiten von Benutzern können im Zeitverlauf Änderungen unterworfen sein. Um zu vermeiden, dass aus diesen organisatorischen Änderungen auch Änderungen am Code von IT-Systemen folgen müssen, wird im Approval Workflow eine rollenbasierte Zuordnung von Akteuren angewendet.
- **Approval Workflow -> Interpreter** : Interpretation der Workflow-Regeln
  Das Approval-Workflow-Pattern besagt, dass der Ablauf eines Approval Workflow durch eine, aus formalen Regeln bestehende Ablaufdefinition zu spezifizieren ist. Um diese Regeln in einem konkreten System anzuwenden, ist ein geeigneter Interpreter notwendig. Der Interpreter liest die Regeln und bringt sie anschließend zur Ausführung.
- **Approval Workflow -> Data Access Object (DAO)** : Speicherung der Konfigurationsdaten
  Um den Zugriff auf die Konfigurationsdaten (Regeln) zu vereinfachen ist eine Kombination mit dem DAO-Pattern sinnvoll.
- **Approval Workflow -> Roles** : Vereinfachung des Bestätigungs- oder Freigabeworkflow
  Wird vor der Verwendung von Daten zu deren Validierung ein Freigabeworkflow eingesetzt, so kann die Definition der Freigabebedingungen durch die Verwendung von Rollen, die jeweils eine Menge von Benutzer repräsentieren, deutlich vereinfacht werden.
- **Validation Requirements -> Eingabevalidierung** : Bereich: Dateneingabe
  Geerbte Beziehung: fuehrt_zu_beziehung
  Ein Bereich der Validation Requirements befasst sich mit der Validierung von Daten während deren Eingabe bzw. Übertragung an das System, in dem die Validierung durchgeführt wird.
- **Approval Requirements -> Eingabevalidierung** : Anforderungen für spezielle Eingabevalidierung
  Geerbte Beziehung: fuehrt_zu_beziehung

Die Nutzung eines Approval Workflow, also der Bestätigung von eingegebenen Werten durch einen weiteren Benutzer (i.d.R. einen fachlichen Experten), ist eine spezielle Form der Eingabevalidierung.

## *15. Asset Valuation*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 103

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

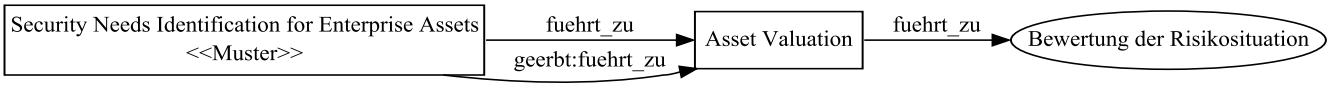
Gruppe(n)

- Analyse der Sicherheitssituation

Kurzbeschreibung

Das Pattern "Asset Valuation" beschreibt ein Verfahren zur Bewertung der Wichtigkeit von Gütern, Produkten, Leistungen und Daten, die in direktem oder indirektem Bezug zu einem bestehenden oder zukünftigen Softwaresystem stehen. Die Wichtigkeit wird dabei auf Basis des Schadens bestimmt, der durch die Zerstörung, Funktionsunfähigkeit, Veröffentlichung usw. des jeweiligen Assets entstehen kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Security Needs Identification for Enterprise Assets -> Asset Valuation** :
  Bewertung bestehender Infrastruktur
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Asset Valuation -> Bewertung der Risikosituation** : Input: bewertete bestehende Systeme und Inhalte
  Quelle: S. 60 und 61 in Security Patterns, Schumacher et al.: Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
- **Security Needs Identification for Enterprise Assets -> Analyse der Sicherheitssituation** : Grobanalyse zu Detailanalyse

Geerbte Beziehung: fuehrt_zu_beziehung
Das Security-Needs-Identification-for-Enterprise-Assets-Pattern dient der Identifikation und Bewertung gefährdeter Güter, Anlagen, Personen und Daten in den untersuchten medizinischen Versorgungseinrichtungen. Die ermittelten Gefährdeten, sowie das ebenfalls ermittelte Einflusspotenzial auf den Gesamtbetrieb dienen dazu im Rahmen der Analyse der Sicherheitssituation ermittelte Sicherheitsrisiken und andere Einflussfaktoren nach ihrer Wichtigkeit zu bewerten.

## *16. Attribute based Access Control*

Quellen

**Paper: ABAC - Ein Referenzmodell für attributbasierte Zugriffskontrolle, Priebe**

Torsten Priebe u. a., "ABAC - Ein Referenzmodell für attributbasierte Zugriffskontrolle," in Sicherheit, 2005, 285-296. http://subs.emis.de/LNI/Proceedings/Proceedings62/GI-Proceedings.62-30.pdf

Schwerpunkt(e)

- Sicherheit

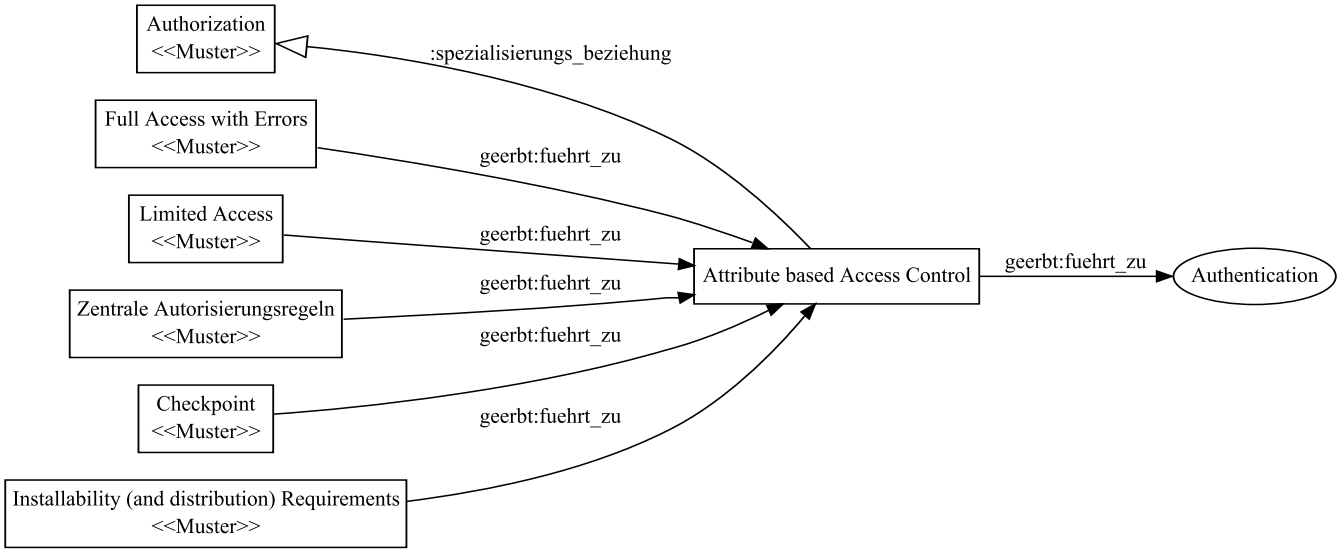
Gruppe(n)

- Access Control or Authorization

Kurzbeschreibung

Das Pattern "Attribute based Access Control" beschreibt ein Verfahren, das die Zugriffsrechte zwischen Benutzern und Objekten dynamisch auf Basis der Attribute des den Benutzer beschreibenden Objekts sowie der zu sichernden Objekte ermittelt.

Beziehungen - Grafik



Beziehungen - Detail

- **Authorization -> Attribute based Access Control** : Authorisierung auf Basis verschiedenster Attribute
  Das Attribute based Access Control Pattern ist eine spezielle Form der Autorisierung. Es beschreibt gemäß http://subs.emis.de/LNI/Proceedings/Proceedings62/GI-Proceedings.62-30.pdf, Abbildung 1, die Verwaltung von Zugriffsrechten auf Objekte durch Subjekte auf Basis von Attributen des Objekts.
- **Full Access with Errors -> Access Control or Authorization** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.
- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Geerbte Beziehung: fuehrt_zu_beziehung
  Aktenbezogene Autorisierungskonzepte lassen sich durch Kombination von Patterns der Gruppe Access Control or Authorization umsetzen.
  Bei der Verwendung zentraler Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.
- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al., S. 287;
  *"..., a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*
- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts

notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.

## 17. Audit Requirements

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 369 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

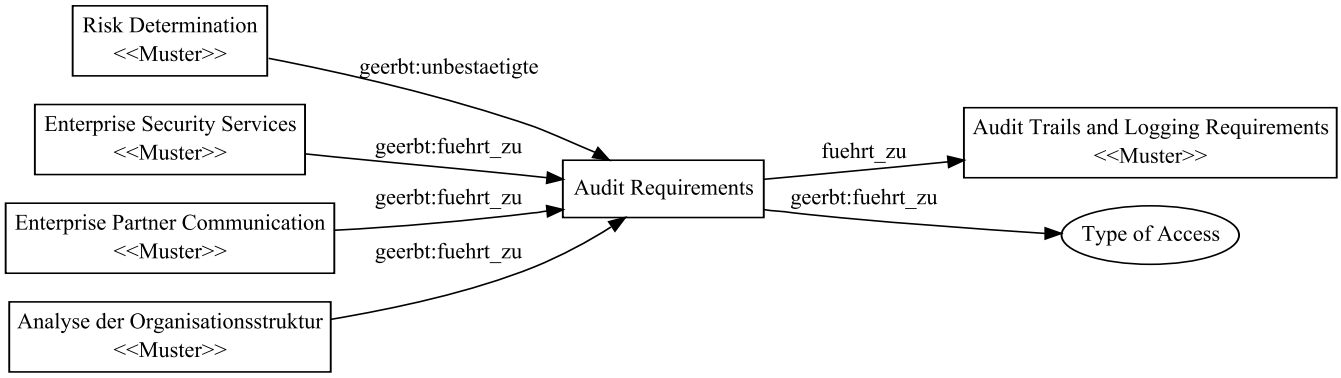
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "Audit Requirements" dient der Spezifikation von Anforderungen an die Nachvollziehbarkeit zurückliegender Systemereignisse, also der Auditierbarkeit des Systems bezüglich bestimmter Ereignisse. Dazu bietet es ein Menge allgemeiner Anforderungen und eine Beschreibung des zugehörigen Prozesses der Ermittlung von Auditierbarkeits-Anforderungen.

Beziehungen - Grafik




Beziehungen - Detail

- **Audit Requirements -> Audit Trails and Logging Requirements** : Definition von Prüfpaden und Logging
  Das Audit-Trails-and-Logging-Requirements-Pattern kann nach der Anwendung des Audit-Requirements-Pattern angewendet werden. Es liefert für die Anwendung des Audit-Trails-and-Logging-Requirements-Patterns die notwendigen Informationen über

die Audit Requirements und ihre relative Wichtigkeit. Diese Beziehung wird in den folgenden beiden Stellen der Quelle "Security Patterns, Schumacher et al." erwähnt:
    - S. 377 im Abschnitt See also
    - S. 378 im Abschnitt Context
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.
- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** : Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: *„Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*(übersetzt aus dem Englischen)
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit, zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** : Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** : Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access

Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *18. Audit Trails and Logging Requirements*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 378 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

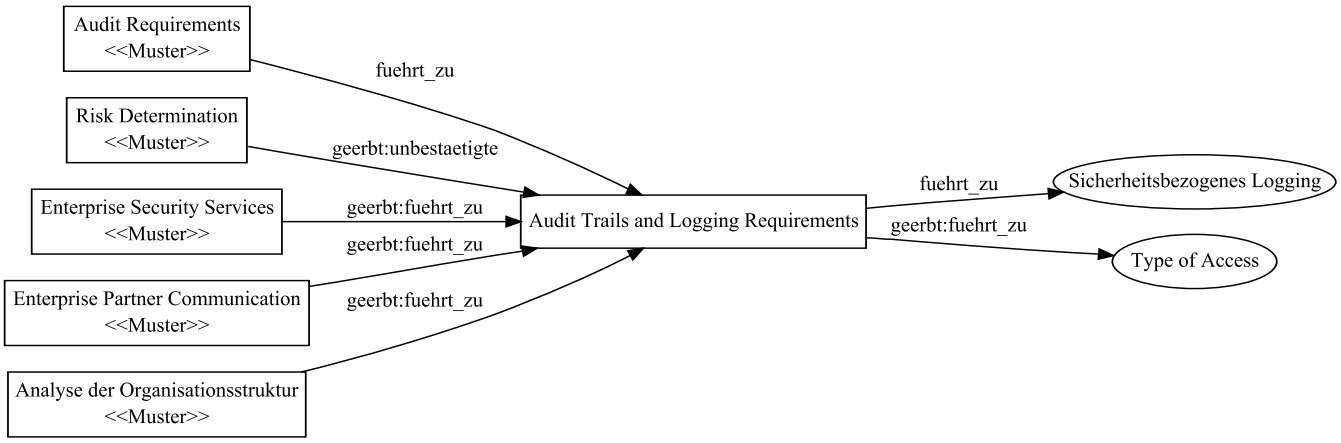
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "Audit Trails and Logging Requirements" lässt sich übersetzt als "Protokollierungs-Anforderungen" übersetzen. Es setzt voraus, dass das Pattern "Audit Requirements" bereits angewendet worden ist, und dient der Spezifikation, wie die Protokollierungsdaten erfasst und gespeichert werden sollen.

Beziehungen - Grafik



Beziehungen - Detail

- **Audit Requirements -> Audit Trails and Logging Requirements** : Definition von Prüfpaden und Logging
  Das Audit-Trails-and-Logging-Requirements-Pattern kann nach der Anwendung des Audit-Requirements-Pattern angewendet werden. Es liefert für die Anwendung des Audit-Trails-and-Logging-Requirements-Patterns die notwendigen Informationen über die Audit Requirements und ihre relative Wichtigkeit. Diese Beziehung wird in den folgenden beiden Stellen der Quelle "Security Patterns, Schumacher et al." erwähnt:
  - S. 377 im Abschnitt See also
  - S. 378 im Abschnitt Context
- **Audit Trails and Logging Requirements -> Sicherheitsbezogenes Logging** : umsetzbar durch
  Die Gruppe sicherheitsbezogenes Logging sollte spezielle Patterns für die Umsetzung von Logging-Mechanismen enthalten. Hierbei handelt es sich um einen offenen Punkt zur Vervollständigung der Pattern-Sprache. Durch die Beziehung zu dieser leeren Gruppe soll die Notwendigkeit zur Vervollständigung dokumentiert werden.
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.
- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** : Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: *„Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“* (übersetzt aus dem Englischen)
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche

Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.

- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *19. Authenticator*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 323 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

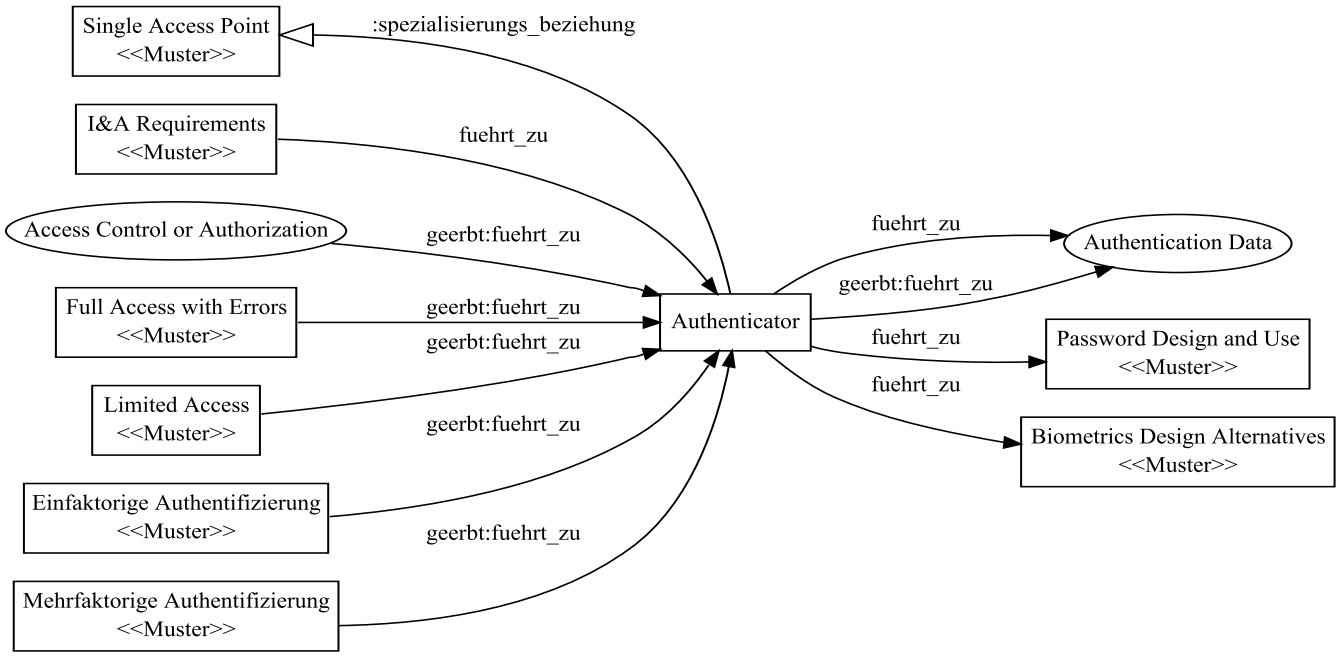
Gruppe(n)

- Authentication

Kurzbeschreibung

Das Pattern "Authenticator" befasst sich auf relativ abstrakter und von konkreten Authentifizierungsmechnismen unabhängiger Weise mit der Frage, wie die Identität eines mit dem System interagierenden Subjekts zuverlässig nachgewiesen werden kann.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Single Access Point -> Authenticator** : Spezialform
  Quelle: Security Patterns, Schumacher et al; S. 327
  *"Single access point is an abstract pattern applied here: Authenticator is a concrete application of it."*
- **I&A Requirements -> Authenticator** : Anforderungen an Authentifizierung
  Zur Umsetzung des Authenticator-Patterns werden Authentifizierungsanforderungen wie z. B. die Anzahl der Benutzer oder der Grad an benötigter Sicherheit vorausgesetzt. Siehe S. 325, Security Patterns, Schumacher et al.
- **Authenticator -> Authentication Data** : Speicherung bzw. Verwaltung der Authentifizierungsdaten
  Zur Authentifizierung/Authentisiserung werden Daten benötigt, die ein Subjekt eindeutig identifizieren und solche, die es dem Subjekt möglich machen, nachzuweisen, ein bestimmtes Subjekt zu sein. Diese Daten zu Subjekten müssen verwaltet werden.
- **Authenticator -> Password Design and Use** : wenn Benutzername/Passwort verwendet wird
  Wenn Passwörter zur Authentifizierung verwendet werden, so führt die Anwendung des Authenticator-Patterns zur Verwendung des Password-Design-and-Use-Patterns. Quelle der Beziehung: S. 327, Security Patterns, Schumacher et al.
- **Authenticator -> Biometrics Design Alternatives** : wenn ein biometrisches Credential verwendet wird
  Diese Beziehung leitet sich aus der Beziehung zwischen Authenticator und Password Design and Use ab. Wenn im Authenticator ein biometrisches Credential verwendet wird, so nimmt das Biometrics-Design-Alternatives-Pattern die Rolle des Password-

Design-and-Use-Patterns ein.
Beziehung ist abgeleitet von: S. 327, Security Patterns, Schumacher et al.

- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.
- **Full Access with Errors -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Full Access with Errors beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
  Quelle: Security Patterns, Schumacher et al.; S: 309
  Beschreibt die Beziehung zu Identifcation und Authentication Patterns.
- **Limited Access -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Limited Access beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
- **Einfaktorige Authentifizierung -> Authentication** : Authentifizierung mit einfachem Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Einfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von nur einem Authentifizierungsverfahren. Typische Beispiele sind:
  - Zugang zum System durch Benutzername/Passwort
  - Zugangssicherung durch Fingerabdruck-Scan
- **Mehrfaktorige Authentifizierung -> Authentication** : komplexes Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Mehrfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von mehr als einem Authentifizierungsverfahren. Die mindestens zwei verschiedenen Authentifizierungsverfahren sollen verschiedenartige Credentials aufweisen.
  Beispiel: Passwort (Wissen) + RSA-Token-ID (Besitz).
- **Authentication -> Authentication Data** : als Datenquelle
  Geerbte Beziehung: fuehrt_zu_beziehung
  Für die Authentifizierung von Benutzern oder Geräten werden Authentifizierungsdaten z. B. in Form von Credentials benötigt.

## *20. Authorization*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann,**

**Sommerlad**

S. 245 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

## Schwerpunkt(e)

- Sicherheit

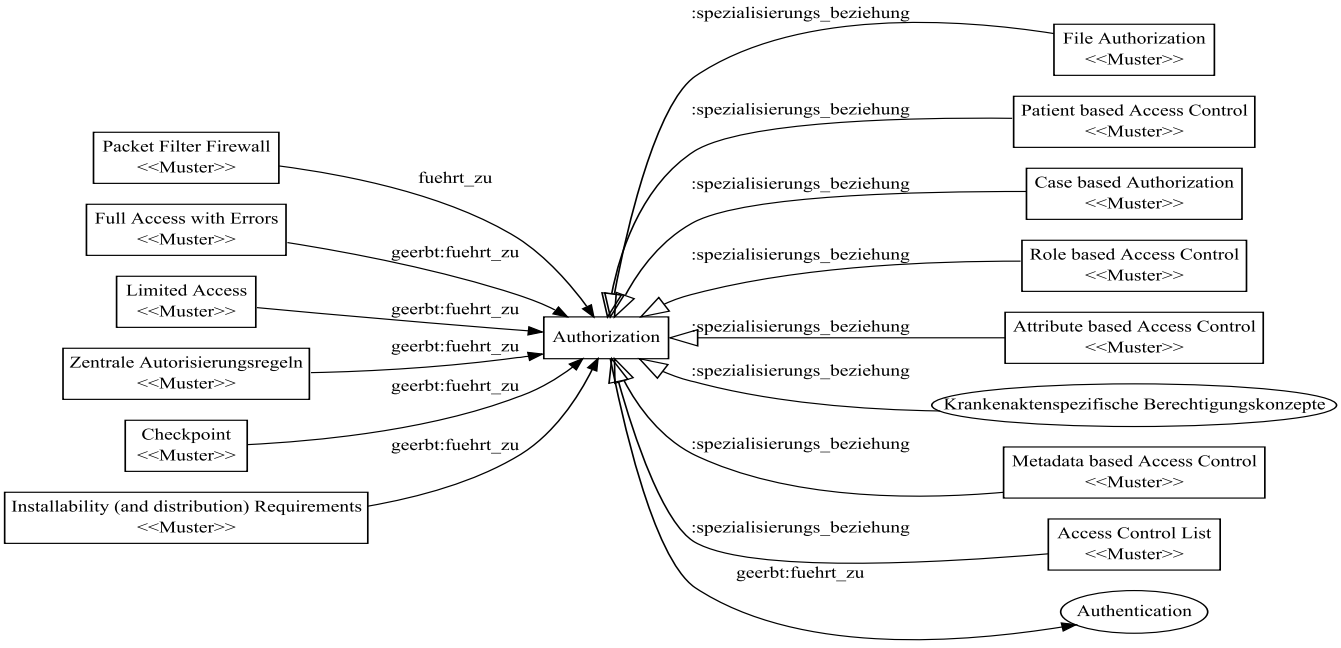
## Gruppe(n)

- Access Control or Authorization

## Kurzbeschreibung

Das Pattern "Authorization" befasst sich auf relativ abstrakter und von konkreten Mechnismen unabhängiger Weise mit der Frage, wie Zugriffsberechtigungen auf Objekte und Bestandteile der Infrastruktur gesteuert werden können.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Packet Filter Firewall -> Authorization** :
  Regelbasierte Authorisierung von Paketen.
  Quelle: Security Patterns; Schumacher et al.; S. 410
- **Authorization -> File Authorization** : Granularität: Einzelne Datei
  Quelle: Security Patterns; Schumacher et al.; S. 351; Absätze Solution und Structure

- **Authorization -> Patient based Access Control** : Granularität: Einzelner Patient
  Diese Beziehung ist abgeleitet von der belegten Beziehung zwischen Authorization und Role based Access Control.
- **Authorization -> Case based Authorization** : Granularität: Einzelner Behandlungsfall
  Diese Beziehung ist abgeleitet von der belegten Beziehung zwischen Authorization und Role based Access Control.
- **Authorization -> Role based Access Control** : Rechte durch Rollen
  Das Role-based-Access-Control-Pattern ist eines Spezialisierung des Authorization-Patterns. (Quelle: S. 248, Security Patterns, Schumacher et al.)
- **Authorization -> Attribute based Access Control** : Authorisierung auf Basis verschiedenster Attribute
  Das Attribute based Access Control Pattern ist eine spezielle Form der Autorisierung. Es beschreibt gemäß http://subs.emis.de/LNI/Proceedings/Proceedings62/GI-Proceedings.62-30.pdf, Abbildung 1, die Verwaltung von Zugriffsrechten auf Objekte durch Subjekte auf Basis von Attributen des Objekts.
- **Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Die Gruppe Krankenaktenspezifische Berechtigungskonzepte beinhaltet eine Menge von Patterns die spezifisch für den Einsatz in verteilten Krankenakten und eine Spezialisierung des Authorization-Patterns sind. (z. B. Case based Authorization)
- **Full Access with Errors -> Access Control or Authorization** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.
- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Geerbte Beziehung: fuehrt_zu_beziehung
  Aktenbezogene Autorisierungskonzepte lassen sich durch Kombination von Patterns der Gruppe Access Control or Authorization umsetzen.
  Bei der Verwendung zentraler Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.
- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al., S. 287;

*"..., a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*

- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.

## *21. Automated I&A Design Alternatives*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 207

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

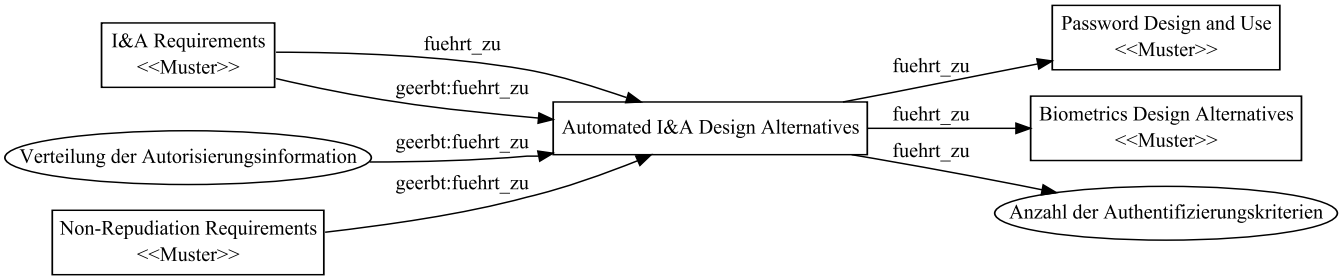
Gruppe(n)

- Grundlegende Authentifizierungskonzepte

Kurzbeschreibung

Das Pattern "Automated I&A Design Alternatives" beschreibt alternative Techniken für die automatisierte Identifikation und Autorisierung von Subjekten und hilft so, für den konkreten Anwendungsfall geeignete Techniken zu finden.

Beziehungen – Grafik



Beziehungen - Detail

- **I&A Requirements -> Automated I&A Design Alternatives** : dienen als Anforderungen für
  Siehe S. 207, Security Patterns, Schumacher et al.
- **Automated I&A Design Alternatives -> Password Design and Use** :
  I&A Methode = Passwort
  Quelle: Security Patterns; Schumacher et al.; S. 209; 2. Satz im Abschnitt Solution.
- **Automated I&A Design Alternatives -> Biometrics Design Alternatives** : I&A Methode = Biometrie
  Quelle: Security Patterns; Schumacher et al.; S. 229
- **Automated I&A Design Alternatives -> Anzahl der Authentifizierungskriterien** : Anzahl der Authentifizierungskriterien
  Je nach Sicherheitsbedarf der Applikation muss ein Subjekt ein oder mehrere Credentials richtig befüllen können um sich erfolgreich zu Authentisieren.
- **Verteilung der Autorisierungsinformation -> Grundlegende Authentifizierungskonzepte** : Benutzeridentifikation
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um die Gültigkeit von Autorisierungsbedingungen bei einem Funktionsaufruf prüfen zu können, ist die Identifikation des zu Berechtigenden mittels Authentifizierung notwendig.
- **I&A Requirements -> Grundlegende Authentifizierungskonzepte** : zur Architekturauswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ergebnisse der Anwendung des I&A-Requirements-Patterns bestimmen die Auswahl einer geeigneten I&A Lösung mittels des Automated-I&A-Design-Alternatives-Patterns, das Bestandteil der Gruppe grundlegende Authentifizierungskonzepte ist. Die Auswahl an Mechanismen erstreckt sich über eine Menge von Authentifizierungsverfahren (Benutzername/Passwort, Biometrie), zentrale- oder verteilte Verwaltung der Authentifizierungsinformation und der Verwendung von einem oder gleichzeitig mehreren Authentifizierungsverfahren.
- **Non-Repudiation Requirements -> Grundlegende Authentifizierungskonzepte** : Authentifizierungsanforderungen für Nichtabstreitbarkeit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns; Schumacher et al.; S. 397; Abbildung "Common model for non-repudiation".

## *22. Availability Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 217 ff.

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation
- Zuverlässigkeit

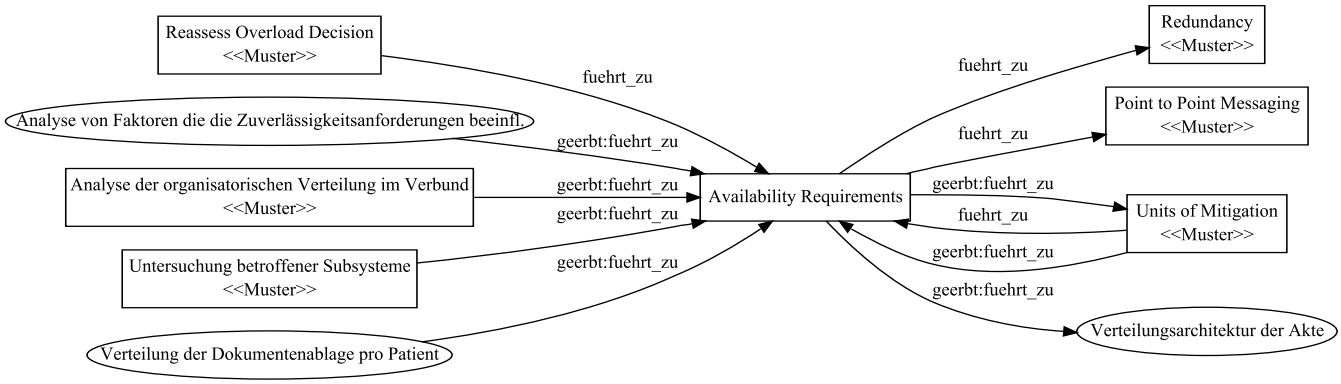
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen
- Kommunikationsanforderungen

Kurzbeschreibung

Das Pattern "Availability Requirements" beschreibt ein Verfahren zur Spezifikation von Anforderungen an die Verfügbarkeit des Systems und/oder seiner einzelnen Bestandteile.

Beziehungen - Grafik




Beziehungen - Detail

- **Units of Mitigation -> Availability Requirements** : Einheiten für die Verfügbarkeitsanf. def. werden
  Durch die Anwendung des Units-of-Mitigation-Patterns wird das System in logische Einheiten für die Fehlerkompensation gegliedert. Jede dieser Einheiten trägt durch seine eigene Verfügbarkeit bzw. Fehlertoleranz zur Verfügbarkeit des Gesamtsystems bei.
- **Reassess Overload Decision -> Availability Requirements** : Aktualisieren der Anforderungen
  Das Reassess-Overload-Decision-Pattern behandelt die Anpassung der Auswahl von Mitteln zur Behandlung von durch Überlastung des Systems eingetretenen Problemen. Die Mittel zur Behandlung von Überlast-Situationen werden auf Basis der Verfügbarkeitsanforderungen (Availability Requirements) ausgewählt und umgesetzt. Deshalb müssen die Verfügbarkeitsanforderungen im Fall veränderter Overload Decisions angepasst werden.
- **Availability Requirements -> Redundancy** : umsetzbar durch
  Verfügbarkeitsanforderungen können unter anderem durch die Verwendung redundanter Komponenten erreicht werden. Sollten die Verfügbarkeitsanforderungen durch reduzierte Update-Dauer, Online-Backups usw. nicht erreicht werden können,

so können durch eine redundante Auslegung des Systems einige der Aktionen, die ohne redundante Auslegung zu einer Downtime führen würden, im Hintergrund ausgeführt werden.
Siehe S. 231, Software Requirement Patterns, Withall; Punkt: Replicate Hardware und S. 230, Software Requirement Patterns, Withall; Punkt: Machine shutdown without interrupting system; (nur durch Redundanz erreichbar)

- **Availability Requirements -> Point to Point Messaging** : zuverlässige Nachrichtenübermittlung
  Durch die Anwendung von (transaktionalem) Point to Point Messaging über eine hochverfügbare Infrastruktur zur Nachrichtenübertragung können kürzere Ausfallzeiten einzelner am Gesamtsystem beteiligter Dienste oder Server kompensiert werden.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung

Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.

- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *23. Benutzerverwaltung mit dezentralen Daten*

Englische Bezeichnung

- User Administration with decentralized Data Storage

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Anzahl der Authentifizierungsquellen

Kontext

In einem System aus mehreren beteiligten Knoten sollen Daten und Funktionen gemeinsam genutzt werden. Für die Nutzung der Systembestandteile ist eine Autorisierung notwendig. Aus diesem Grund müssen die beteiligten Benutzer authentifiziert werden.

Problem
Die zur Authentifizierung notwendigen Daten sind in den verschiedenen Knoten und nicht zentral in einem allen Knoten gemeinsamen Authentifizierungssystem gespeichert. Dieser Sachverhalt erschwert die Nutzung des verteilten Gesamtsystems als kohärentes System, das gegenüber dem Benutzer verteilungstransparent sein soll. Eine Zusammenführung der zur Authentifizierung notwendigen Daten in eine Benutzerverwaltung mit zentral gespeicherten Daten ist aus organisatorischen oder anderen Gründen nicht möglich.

Lösung
Jedes Subsystem verwaltet seine Benutzer selbst. Gegebenenfalls müssen Benutzer, die mehrere der Subsysteme benutzen in jedem beteiligten Subsystem separat erfasst werden.

Vorteile:

- Kompatibel mit wenig flexiblen Systemkonten (z. B. Krankenhausinformationssystemen, die keine konfigurierbaren Benutzerdatenquellen unterstützen)
- Keine Veränderungen an der Benutzerverwaltung bestehender Primärsysteme der an einer verteilten Krankenakte beteiligten Organisationen notwendig.

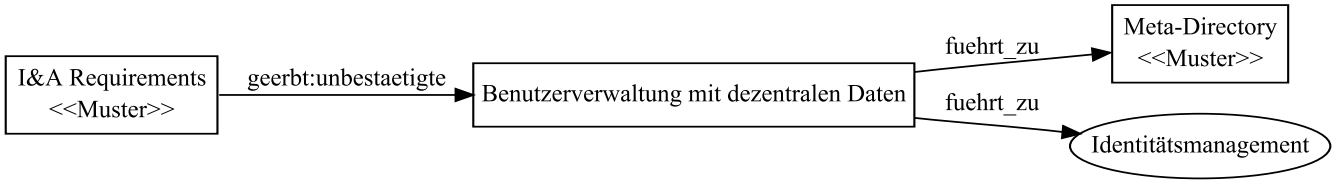
Nachteile:

- Berechtigungen von Benutzern, die nicht im Primärsystem gepflegt sind, können nicht direkt im Primärsystem, das der Ort der Speicherung von z. B. Dokumenten ist, festgelegt werden.
- Gleiche natürliche Personen können Benutzer mit unterschiedlichen Berechtigungen in den verschiedenen Primärsystemen haben.
- Das Nachvollziehen von Berechtigungen eines Benutzers im Gesamtsystem ist nur schwer möglich.

Möglichkeiten zur Kompensation von Nachteilen

Eine Kombination der Benutzerverwaltung mit dezentral gespeicherten Daten mit dem Meta-Directory-Pattern ermöglicht eine Integration von Authentifizierungsinformation zu einem virtuell zentralen Authentifizierungsdienst.

Beziehungen - Grafik




Beziehungen - Detail

- **Benutzerverwaltung mit dezentralen Daten -> Meta-Directory** : Benötigt zur Erreichung einheitlicher Benutzer
  Eine Kombination der Benutzerverwaltung mit dezentral gespeicherten Daten mit dem Meta-Directory-Pattern ermöglicht eine Integration von Authentifizierungsdaten-quellen zu einem virtuell zentralen Authentifizierungsdienst.
- **Benutzerverwaltung mit dezentralen Daten -> Identitätsmanagement** : Zur Integration unterschiedlicher Benutzerstämme
  Das Pattern Benutzerverwaltung mit dezentralen Daten beschreibt einen Lösungsansatz der darauf basiert, dass jede Organisation ihre Benutzer selbst verwaltet und selbst in einem oder mehreren eigenen Systemen zur Benutzerverwaltung speichert. Das ist unter anderem der Fall, wenn verschiedene Primärsysteme mit jeweils eigener Benutzerverwaltung verwendet werden. Sollen derartige Systeme zu einem komplexen Systemverbund integriert werden, so ist die Verwendung von Patterns der Gruppe Identitätsmanagement zwangsläufig.

- **I&A Requirements -> Anzahl der Authentifizierungsquellen** : Beeinflusst die Auswahl
  Geerbte Beziehung: unbestaetigte_beziehung

## *24. Biometrics Design Alternatives*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 229 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

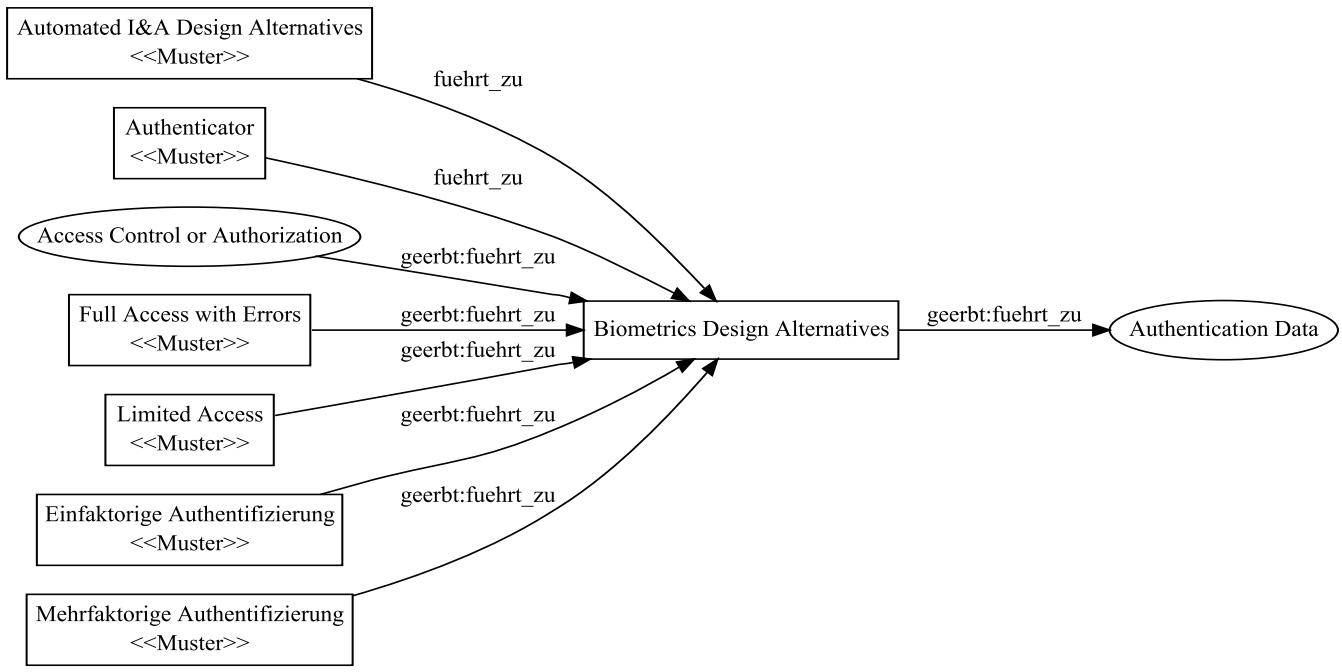
Gruppe(n)

- Authentication

Kurzbeschreibung

Dieses Muster hilft bei der Auswahl geeigneter biometrische Mechanismen zur Erfüllung konkreter Anforderungen an die Qualität der Identifikation und Authentifizierung von Personen.

Beziehungen - Grafik



Beziehungen - Detail

- **Automated I&A Design Alternatives -> Biometrics Design Alternatives** : I&A Methode = Biometrie
  Quelle: Security Patterns; Schumacher et al.; S. 229
- **Authenticator -> Biometrics Design Alternatives** : wenn ein biometrisches Credential verwendet wird
  Diese Beziehung leitet sich aus der Beziehung zwischen Authenticator und Password Design and Use ab. Wenn im Authenticator ein biometrisches Credential verwendet wird, so nimmt das Biometrics-Design-Alternatives-Pattern die Rolle des Password-Design-and-Use-Patterns ein.
  Beziehung ist abgeleitet von: S. 327, Security Patterns, Schumacher et al.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.
- **Full Access with Errors -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Full Access with Errors beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
  Quelle: Security Patterns, Schumacher et al.; S: 309
  Beschreibt die Beziehung zu Identifcation und Authentication Patterns.
- **Limited Access -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Limited Access beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
- **Einfaktorige Authentifizierung -> Authentication** : Authentifizierung mit einfachem Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Einfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von nur einem Authentifizierungsverfahren. Typische Beispiele sind:
  - Zugang zum System durch Benutzername/Passwort
  - Zugangssicherung durch Fingerabdruck-Scan
- **Mehrfaktorige Authentifizierung -> Authentication** : komplexes Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Mehrfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von mehr als einem Authentifizierungsverfahren. Die mindestens zwei verschiedenen

Authentifizierungsverfahren sollen verschiedenartige Credentials aufweisen.
Beispiel: Passwort (Wissen) + RSA-Token-ID (Besitz).

- **Authentication -> Authentication Data** : als Datenquelle
  Geerbte Beziehung: fuehrt_zu_beziehung
  Für die Authentifizierung von Benutzern oder Geräten werden Authentifizierungsdaten z. B. in Form von Credentials benötigt.

## *25. Bridge*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 151 ff.

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

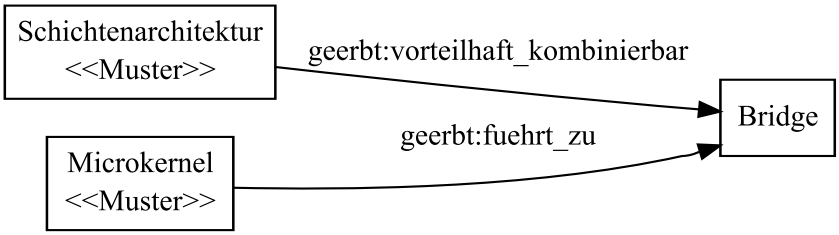
Gruppe(n)

- Handle Body Patterns
- Kopplung und Entkopplung

Kurzbeschreibung

Das Pattern "Bridge" entkoppelt eine Abstraktion von ihrer konkreten Implementierung, sodass beide voneinander unabhängig variiert werden können. Diese Variation kann sowohl seitens der Abstraktion als auch seitens der Implementierung mittels Vererbung erreicht werden.

Beziehungen - Grafik




Beziehungen - Detail

- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar.

Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.

- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *26. Broker*

Quellen

**POSA 1, A System of Patterns, Buschmann**

S. 99 ff.

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 237ff.

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Kommunikation
- Flexibilität

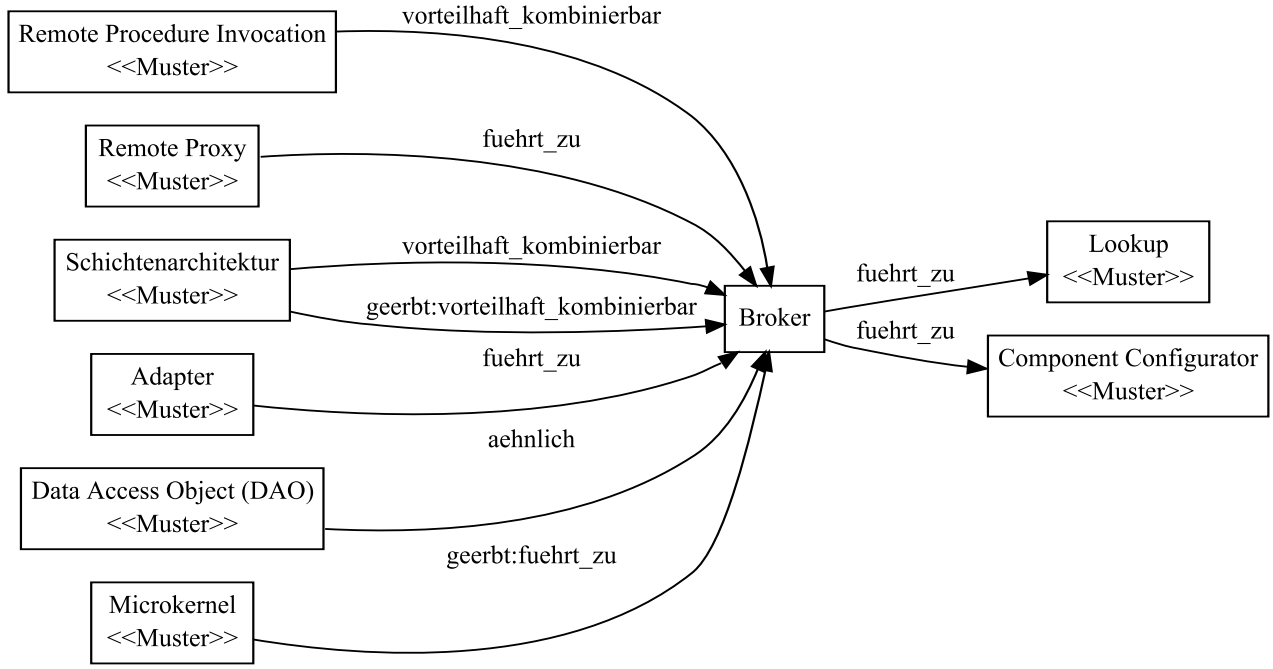
Gruppe(n)

- Kopplung und Entkopplung
- Verbergen der Kommunikation

Kurzbeschreibung

Das Broker-Pattern wird im Rahmen der Kommunikation zwischen den Bestandteilen eines verteilten Systems eingesetzt. Es kapselt den Aspekt der Kommunikation und verbirgt so den dazu notwendigen Code vor dem Autor des die Anwendungslogik betreffenden Codes.

Beziehungen - Grafik




Beziehungen - Detail

- **Remote Procedure Invocation -> Broker** : Entkopplung
  Zur Entkopplung von Server und Client kann eine Broker-Komponente bzw. Broker-Schicht eingeführt werden, die zwischen den beiden vermittelt. Siehe S. 99 in Pattern Oriented Software Architecture, Volume 1
- **Remote Proxy -> Broker** : Zur Vereinfachung der Proxy-Methoden
  Pattern Oriented Software Architecture, Volume 4 beschreibt auf S. 237, untere Abbildung, die Verwendung von Brokern auf Client- und Serverseite zur Vereinfachung der Kommunikation. Dabei wird der Code für die eigentliche Kommunikation aus den einzelnen Proxy-Methoden in den Broker der Clientseite verlagert.
- **Schichtenarchitektur -> Broker** :
  Siehe Pattern Oriented Software Architecture, Volume 4, S. 238 : *"Most Broker Realizations are based on a Layers Architecture..."*
- **Adapter -> Broker** : Wenn Adaptee Remote
  Wenn das anzupassende Objekt (Adaptee) sich auf einem entfernten System befindet, ist es notwendig, die Adapter-Implementierung mit dem Remote-Proxy-Pattern oder dem Broker-Pattern zu kombinieren.
- **Data Access Object (DAO) -> Broker** :
  Quelle:
  http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html;
  Abschnitt "Related Patterns";
  *“The DAO pattern is related to the Broker pattern, which describes approaches for decoupling clients and servers in distributed systems. The DAO pattern more*

*specifically applies this pattern to decouple the resource tier from clients in another tier, such as the business or presentation tier."*

- **Broker -> Lookup** : kann Lookup Funktionalität anbieten
  Lt. Pattern Oriented Software Architecture, Volume 4, S. 239 kann eine Broker-Implementierung zusätzlich Lookup-Funktionalität anbieten, so dass Clients ortstransparent zugreifen können.
- **Broker -> Component Configurator** : Komponentenkonfiguration zur Laufzeit
  **Quelle:** S. 490; Pattern oriented Software Architecture, Volume 4
  Bei der Realisierung einer Broker-Middleware, in der zur Laufzeit flexible Komponentenkonfiguration möglich sein soll, ist die Anwendung des Component-Configurator-Patterns folgerichtig.
  **Quelle:** S. 239; Pattern oriented Software Architecture, Volume 4
  Ein Component Configurator kann in Verbindung mit dem Broker-Muster verwendet werden, um die Requestor und Invoker Implementierung konfigurierbar (z. B. per Konfiguration austauschbar) zu gestalten.
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *27. Builder*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 97-106

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

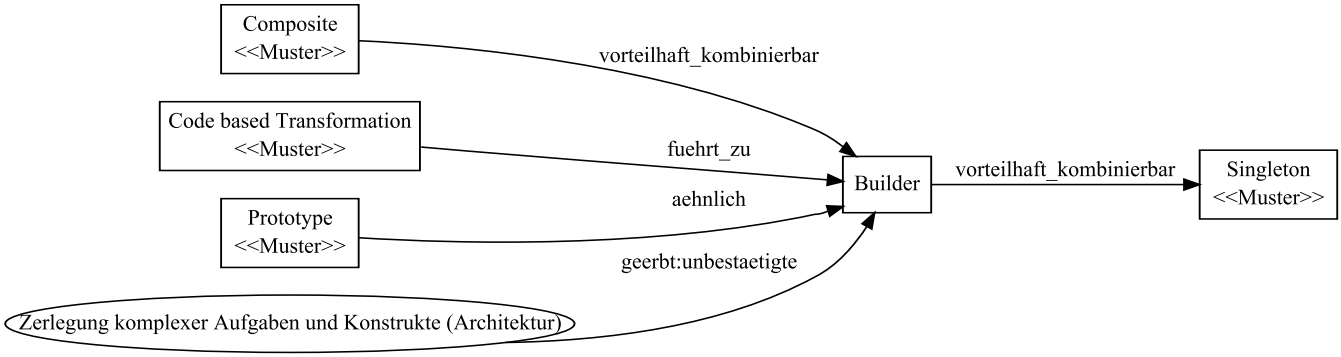
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

## Kurzbeschreibung

Das Builder-Pattern dient der Erstellung komplexer Objekte. Dazu trennt es den Generierungsprozess von der Klasse des komplexen Objekts ab. Bau der Generierungsprozess auf einer Abstraktion der Klasse des Objekts auf, so kann er für verschiedene konkrete, von der Abstraktion abgeleitete Klassen verwendet werden.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Composite -> Builder** : createing composites
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Relationships
  Quelle: Design Patterns; Gamma et al.; S. 106
- **Code based Transformation -> Builder** : Umsetzung unterschiedl. Transformationen
  Zur Entkopplung des Transformationsvorgangs von der eigentlichen Anwendungs- und Übertragungslogik kann die Code based Transformation durch die Anwendung des Builder-Patterns gekapselt werden. Dadurch wird es auch möglich, verschiedene Transformationen durch den gleichen steuernden Code (Director) durchzuführen.
- **Prototype -> Builder** :
  Quelle: Design Patterns; Gamma et al.; S. 119;
  *"Prototype has many of the same consequences that Abstract Factory and Builder have:* [...]*"*
- **Builder -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 134
  *"Many patterns can be implemented using the Singleton pattern. See Abstract Factory (87), Builder(97), and Prototype (117)."*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *28. Business Process Layer*

Quellen

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

Business Process Layer wird auf S. 69 als Bestandteil des Layers (Schichtenarchitektur-Pattern) vorgestellt. Es handelt sich dabei um eine spezielle Schicht. Diese Schicht wiederum kann als eigenes Pattern beschrieben werden.

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Flexibilität

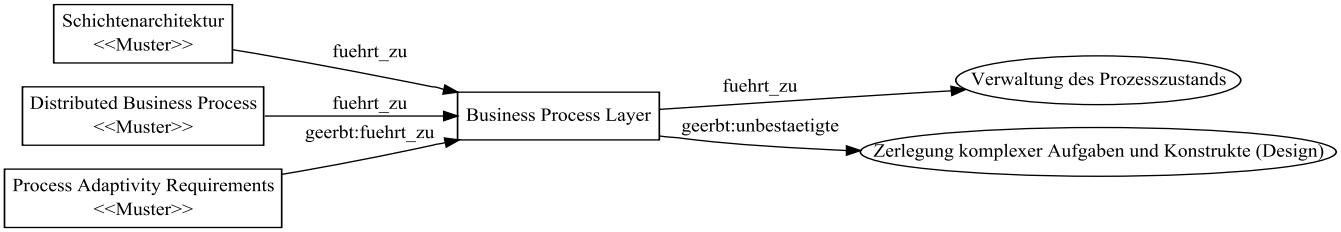
Gruppe(n)

- Prozessabbildung
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kurzbeschreibung

Das Pattern "Business Process Layer" beschreibt den im Bereich serviceorientierter Architekturen verbreiteten Ansatz, der Prozess-Logik eine separate Schicht innerhalb der Schichtenarchitektur der Anwendung einzuräumen. Dadurch lässt sich die Prozess-Logik unabhängig von der Logik der darunterliegenden Schichten anpassen und ändern.

Beziehungen - Grafik




Beziehungen - Detail

- **Schichtenarchitektur -> Business Process Layer** : wenn prozessorientierter Anwendungsbereich
  Bei Anwendungssystemen, deren Aufgabe die Abbildung von einem oder mehreren Prozessen ist, kann durch die Definition einer eigenen Business Process Layer (Geschäftsprozess-Schicht) die Aufgabe der Abbildung des Prozessflusses in einer von anderen Aufgaben der Anwendung entkoppelten Form realisiert werden. Sie eignet sich deshalb besonders für häufiger veränderliche Prozesse.
- **Distributed Business Process -> Business Process Layer** : Bei Schichtenarchitektur Sollen durch ein Anwendungssystem ein oder mehrere Geschäftsprozesse abgebildet werden, die auf mehr als einem Quellsystem basieren, spricht man von einem

verteilten Geschäftsprozess (Distributed Business Process). Sollen verteilte Geschäftsprozesse durch ein gemeinsames Frontend zur Verfügung gestellt werden, das die technisch verteilte Natur des Prozesses gegenüber dem Benutzer transparent macht, so ist zwischen der Benutzeroberfläche und einer Integrationsschicht zur Anbindung der Quellsysteme zusätzlich eine Business Process Layer (siehe POSA4, S. 69 ff.) zweckmäßig.

- **Business Process Layer -> Verwaltung des Prozesszustands** : notwendig zur Abbildung
  Um einen Geschäftsprozess durch Software abbilden zu können, muss das zugehörige IT-System Kenntniss über den Zustand einer jeden Prozess-Instanz besitzen. Auf Basis des Zustands wird vom System der nächste notwendige Bearbeitungsschritt ermittelt. Wie dieser Zustand technisch verwaltet wird kann unterschiedlich gelöst werden. Die Gruppe Verwaltung des Prozesszustands stellt mehrere Kandidaten zur Auswahl.
- **Process Adaptivity Requirements -> Prozessabbildung** : Input: Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Process Adaptivity Requirements sind die Flexibilitätsanforderungen an den gewählten Mechanismus zur Prozessabbildung. Hohe Flexibilitätsanforderungen benötigen umfassende Möglichkeiten zur Konfiguration von Prozessen, niedrige Anforderungen aus relativ statischen Prozessen führen zu stärker direkt im Quellcode des Systems realisierten Prozessen.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *29. Case based Authorization*

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Krankenaktenspezifische Berechtigungskonzepte

### Kontext

Verschiedene medizinische Einrichtungen oder Organisationseinheiten einer Einrichtung wollen bezogen auf einzelne Behandlungsfälle eine gemeinsame Behandlung von Patienten durchführen. Die Vorbedingung, dass Patienten und medizinisches Personal über Einrichtungsgrenzen hinweg im System eindeutig identifiziert werden könne, ist erfüllt.

### Problem

Die gegebene Rechtssituation erlaubt es nicht, allen beteiligten Kräften des medizinischen Personals alle Daten zu einem Patienten einzusehen. Um im Rahmen des jeweils aktuellen Behandlungsfalls die geforderten Leistungen erbringen zu können, benötigen die beteiligten Leistungserbringer üblicherweise Basis-Informationen zu Patienten sowie Informationen aus zuvor erbrachten Leistungen. Eine individuelle Vergabe von Zugriffrechten auf einzelne Dokumente ist allerdings häufig sehr komplex und nur durch geeignete Vorsysteme und Erfassungsprozesse zu erreichen.

### Lösung

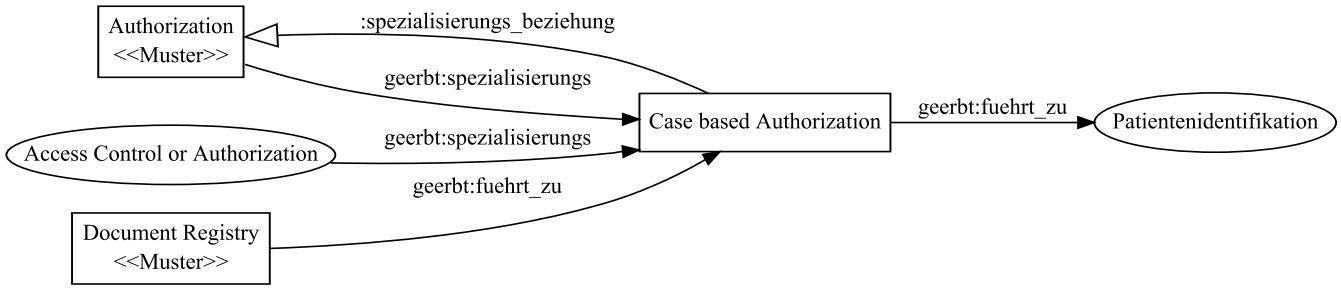
Eine Lösung des beschriebenen Problems ist die Vergabe eines gemeinsamen Rechts auf alle zu einem Behandlungsfall gehörigen Dokumente zuzugreifen. Das bedeutet, alle Organisationseinheiten, die an dem einrichtungsübergreifenden Behandlungsfall beteiligt sind, können auf eine fallbezogene Dokumentensammlung zugreifen.

Die Erteilung von Zugriffsberechtigungen erfolgt auf Basis der Attribute Fall und Organisationseinheit.

### Beispiel

Dieses Pattern basiert auf der Idee der elektronischen Fallakte (http://www.fallakte.de/).

### Beziehungen - Grafik




### Beziehungen - Detail

- **Authorization -> Case based Authorization** : Granularität: Einzelner Behandlungsfall
  Diese Beziehung ist abgeleitet von der belegten Beziehung zwischen Authorization und Role based Access Control.
- **Access Control or Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Krankenaktenspezifische Berechtigungskonzepte sind spezielle

Autorisierungsmechanismen. Sie leiten sich von allgemeinen Autorisierungsmechanismen ab.

- **Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Die Gruppe Krankenaktenspezifische Berechtigungskonzepte beinhaltet eine Menge von Patterns die spezifisch für den Einsatz in verteilten Krankenakten und eine Spezialisierung des Authorization-Patterns sind. (z. B. Case based Authorization)
- **Document Registry -> Krankenaktenspezifische Berechtigungskonzepte** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Kontext medizinischer Informationssysteme ist häufig bereits das Wissen über die Existenz eines Dokuments zu einem bestimmten Typ von Behandlung datenschutzrechtlich relevant. Aus diesem Grund ist es notwendig auch in einem Verzeichnisdienst, der die Suche über verteilt abgelegte Dokumente ermöglicht, ein umfassendes Berechtigungssystem zu integrieren.
- **Krankenaktenspezifische Berechtigungskonzepte -> Patientenidentifikation** :
  Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im medizinischen Kontext ist die zentrale Entität i.d.R. der Patient. Seine eindeutige Identifizierung, auch über Systemgrenzen hinweg, ist die Basis der meisten medizinspezifischen Autorisierungsmechanismen.

## *30. Central Message Bus*

Quellen

**SOA Design Patterns, Erl**

S. 704 (Enterprise Service Bus)
Das Enterprise Service Bus Pattern ist eine speziell auf die Anwendung einer Service orientierten Architektur ausgelegte Ausprägung des Central Message Bus Pattern.

SOA Design Patterns, Thomas Erl, Prentice Hall, 3. Auflage, Februar 2010.

**Enterprise Integration Patterns, Hohpe**

S. 137 ff. (Message Bus)
Das Message Bus Pattern ist die ausschließlich Nachrichtenorientierte Ausprägung des Central Message Bus Pattern.

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Flexibilität

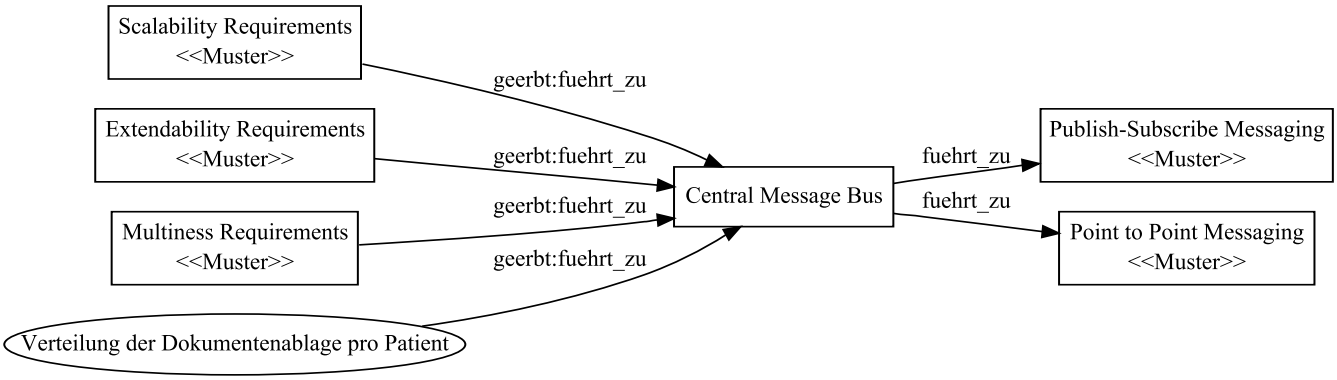
Gruppe(n)

- Grundlegende Architekturstile
- Flexibilitaetserhoehende Architekturmuster

## Kurzbeschreibung

Das Pattern "Central Message Bus" beschreibt allgemein den Ansatz, eine Menge verschiedener Systeme über einen zentralen Kommunikationsknoten zu verbinden. Die Rolle des Central Message Bus kann in einem Krankenhaus z. B. ein HL7-Message-Server oder in einer serviceorientierten Architektur ein Enterprise Service Bus (ESB) einnehmen.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Central Message Bus -> Publish-Subscribe Messaging** : Unterstützt
  Quelle: Enterprise Integration Patterns, Hohpe, S. 139
- **Central Message Bus -> Point to Point Messaging** : Unterstützt
  Point to Point Messaging beschreibt die zuverlässige Übertragung einer Nachricht von einem definierten Sender zu einem ebenfalls definierten Empfänger. Praktisch wird dieses Pattern häufig durch den Einsatz einer Message Queue (vgl. JMS) umgesetzt. Solchen Message Queues kann (siehe Enterprise Integration Patterns, Hohpe, S. 139) ein Message-Routing-Vorgang vorgeschalten sein. Der Central Message Bus kann als zentrale Plattform für die zuverlässige Übermittlung von Nachrichten zwischen Sender und Empfänger dienen.
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.

- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.
- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.

## *31. Chain of Responsibility*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 223-232

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

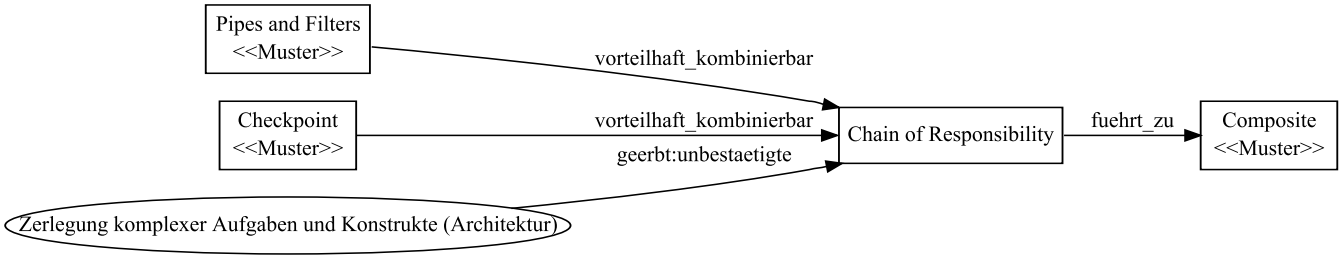
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

Kurzbeschreibung

Das Pattern "Chain of Responsibility" beschreibt das Verfahren, eine nach einem definierten Kriterium geordnete Liste von Routinen an ein Ereignis zu koppeln.

Beziehungen - Grafik



Beziehungen - Detail

- **Pipes and Filters -> Chain of Responsibility** : kann durch realisiert werden
  Das Pipes-and-Filters-Pattern teilt eine Aufgabe in eine Menge sequenzierbarer Arbeitsschritte. Diese Sequenz von Arbeitsschritten kann durch eine Chain of Responsibility abgebildet werden. Eine Chain of Responsibility besteht aus einer verketteten Liste von Handler-Objekten. Diese Handler-Objekte können verwendet werden um je einen Schritt aus dem Pipes and Filters Architektur-Pattern abzubilden.
- **Checkpoint -> Chain of Responsibility** : Delegierung von Entscheidungen
  Quelle: Security Patterns, Schumacher et al., S. 296; *"Check Point implementations can employ Chain of Responsibility to delegate decisions among several concrete Checkpoint implementations."*
  **Beispiele:** PAM unter Linux; JAAS unter Java;
- **Chain of Responsibility -> Composite** : wenn existierende Links verwendet werden
  Siehe Design Patterns, Gamma et al. S. 232:
  Das Chain-of-Responsibility-Pattern wird häufig in Verbindung mit dem Composite-Pattern verwendet. Dabei kann z. B. die Eltern-Komponente als Successor einer Komponente verwendet werden.
  S. 227 der gleichen Quelle beschreibt das als die Ausprägung des Chain-of-Responsibility-Patterns, die existierende Beziehungen zwischen den Objekten verwendet. Davon abweichend existiert auch eine Ausprägung, in der speziell für die Anwendung des Chain-of-Responsibility-Patterns eigene Beziehungen eingebaut werden.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *32. Checkpoint*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 287 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

S. 7

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

### Schwerpunkt(e)

- Sicherheit

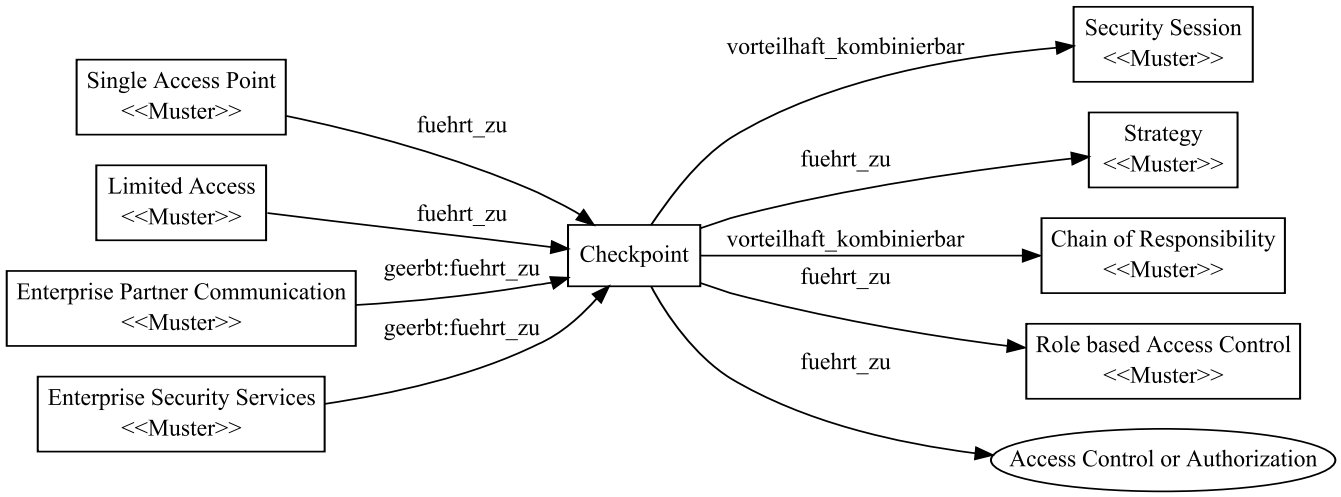
### Gruppe(n)

- Point of Access

### Kurzbeschreibung

Das Pattern "Checkpoint" wird üblicherweise gemeinsam mit dem Pattern "Single Access Point" verwendet. Es beschreibt das Vorgehen, den Single Access Point dazu zu nutzen, die Identität des auf ein System zugreifenden Subjekts und seine Zugriffsberechtigung zu überprüfen.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Single Access Point -> Checkpoint** : Zugangskontrolle
  Quelle: Security Patterns; Schumacher et al; S. 284
  *“Optionally implement the entry check at the single access point.* [...] *Checkpoint shows how to make this checking flexible.“*
- **Limited Access -> Checkpoint** :
  Quelle: Security Patterns; Schumacher et al.; S. 319
  *“Check Point* [...] *should be considered for designing and implementing the user I&A and association of access rights.”*
- **Checkpoint -> Security Session** : typically relies on
  Quelle: Security Patterns; Schumacher et al.; S. 304
  Quelle: S. 299; Security Patterns: *"A systems Checkpoint is the usual Place to instantiate the session object and set up its initial values."*

- **Checkpoint -> Strategy** : Erhöhung der Flexibilität
  Quelle: Security Patterns, Schumacher et al., S. 296;
  *"Check Point uses Strategy for gaining flexibility in application security"*
- **Checkpoint -> Chain of Responsibility** : Delegierung von Entscheidungen
  Quelle: Security Patterns, Schumacher et al., S. 296; *"Check Point implementations can employ Chain of Responsibility to delegate decisions among several concrete Checkpoint implementations."*
  **Beispiele:** PAM unter Linux; JAAS unter Java;
- **Checkpoint -> Role based Access Control** : Sicherheitsprüfungen
  Quelle: Security Patterns, Schumacher et al., S. 296;
  *"Role based Access Control is often used to implement Check Point's Security checks."*
- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Quelle: Security Patterns, Schumacher et al., S. 287;
  *"*[...]*, a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*
- **Enterprise Partner Communication -> Point of Access** : Gestaltung des Zugriffspunkts für Partner
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Enterprise-Partner-Communication-Pattern beschreibt die Kommunkationsanforderungen zwischen Organisationen mit voneinander getrennt existierender IT-Infrastruktur. Zur Umsetzung der Enterprise Partner Communication muss ein Muster für die Gestaltung des Point of Access ausgewählt werden, um für den Kommunikationspartner einen Zugangsweg zu dem betreffenden IT-System zu eröffnen.
- **Enterprise Security Services -> Point of Access** : Gestaltung eines sicheren Zugriffspunkts für interne Applikationen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen sicheren Zugriffsweg auf interne Applikationen zu gestalten sollte ein Pattern der Gruppe Point of Access angewendet werden.

## *33. Checkpoint (Error recovery)*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 166 ff.

Mit Fokus auf "Error Recovery". Inhaltlich abweichend vom, im Bereich Sicherheit gebräuchlichen Checkpoint-Pattern.

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

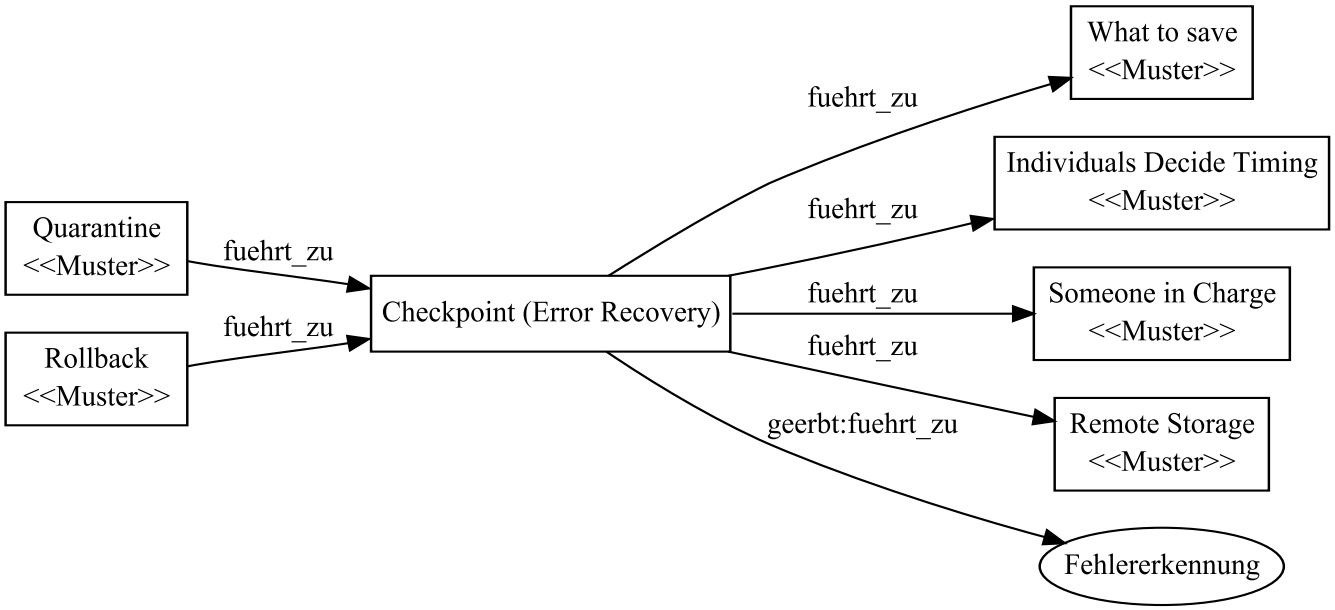
### Gruppe(n)

- Automatisierte Fehlerbehebung

### Kurzbeschreibung

Das Pattern "Checkpoint" aus dem Gebiet der Fehlerbehebung beschreibt das Vorgehen während der Verarbeitung von kritischen Daten zu definierten Zeitpunkten Sicherungen von konsistenten Zuständen anzulegen. Kombiniert mit z. B. dem Pattern "Rollback" kann so auf einen konsistenten Zustand zurückgesetzt werden.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Quarantine -> Checkpoint (Error recovery)** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Rollback -> Checkpoint (Error recovery)** : Benötigt als Bestandteil
  Quelle: Patterns for Fault Tolerant Systems, Hanmer, S. 166;
  *"Many actions that might be taken to isolate or remediate an error will involve rolling the system back to a known place* [dieser bekannte Zustand kann durch die Anwendung des Checkpoint Pattern erreicht werden] *through a Rollback ..."*
- **Checkpoint (Error recovery) -> What to save** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Checkpoint (Error recovery) -> Individuals Decide Timing** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Checkpoint (Error recovery) -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Checkpoint (Error recovery) -> Remote Storage** : Möglicher Speicherort
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 171
  *"What storage location should be used for Checkpoints to reduce the time before execution can be resumed after error recovery?"*
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *34. Checksum*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 129-131

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

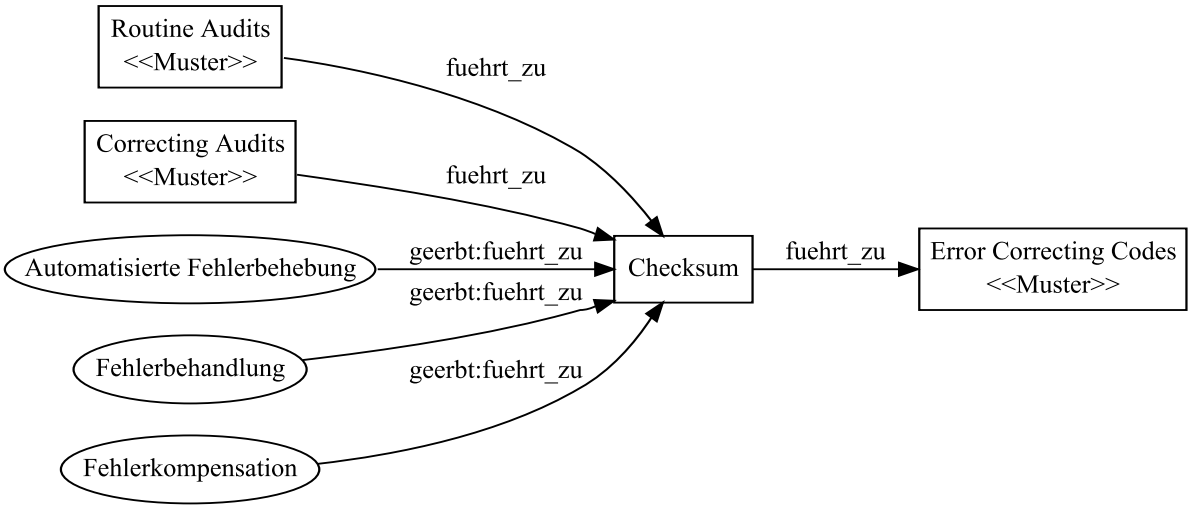
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Checksum" beschreibt das Verfahren Fehler durch eingebaute Prüfsummen Fehler in Daten zu erkennen. Das reicht von einfachen Prüfziffern in Nummern-Codes bis hin zur Übermittlung von Hash-Codes zur Validierung umfangreicher Datenpakete.

Beziehungen - Grafik




Beziehungen - Detail

- **Routine Audits -> Checksum** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Correcting Audits -> Checksum** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Checksum -> Error Correcting Codes** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *35. Circle of Trust*

Quellen

**Paper: A Pattern Language for Identity Management**

A Pattern Language for Identity Management; N. Delessy, E. Fernandez, M. Larrondo-Petrie; Proceedings of the International Multi-Conference on Computing in the Global Information Technology; 2007

### Schwerpunkt(e)

- Sicherheit

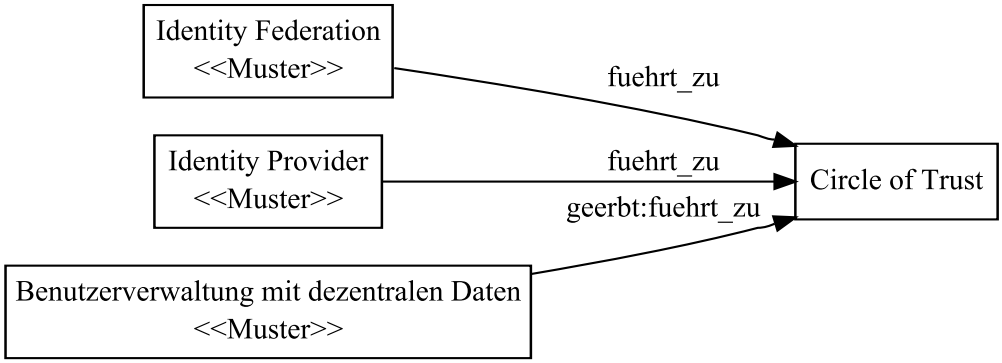
### Gruppe(n)

- Identitätsmanagement

### Kurzbeschreibung

Das Pattern "Circle of Trust" beschreibt den Ansatz, dass verschiedene Dienste-Anbieter einander im bezüglich der Authentifizierung von Benutzern vertrauen, sodass ein Benutzer z. B. nur bei einem der Anbieter innerhalb eines Circle of Trust einen Account benötigt.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Identity Federation -> Circle of Trust** : benutzt
  Quelle: Abbildung 1 (Pattern diagram for identity management) in A Pattern Language for Identity Management (Paper)
- **Identity Provider -> Circle of Trust** : benutzt
  Abbildung 1 (Pattern diagram for identity management) in "A Pattern Language for Identity Management" zeigt die Beziehung.
- **Benutzerverwaltung mit dezentralen Daten -> Identitätsmanagement** : Zur Integration unterschiedlicher Benutzerstämme
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Pattern Benutzerverwaltung mit dezentralen Daten beschreibt einen Lösungsansatz der darauf basiert, dass jede Organisation ihre Benutzer selbst verwaltet und selbst in einem oder mehreren eigenen Systemen zur Benutzerverwaltung speichert. Das ist unter anderem der Fall, wenn verschiedene Primärsysteme mit jeweils eigener Benutzerverwaltung verwendet werden. Sollen

derartige Systeme zu einem komplexen Systemverbund integriert werden, so ist die Verwendung von Patterns der Gruppe Identitätsmanagement zwangsläufig.

## *36. Client-Dispatcher-Server*

Quellen

**POSA 1, A System of Patterns, Buschmann**

S. 323 ff.

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Kommunikation
- Flexibilität

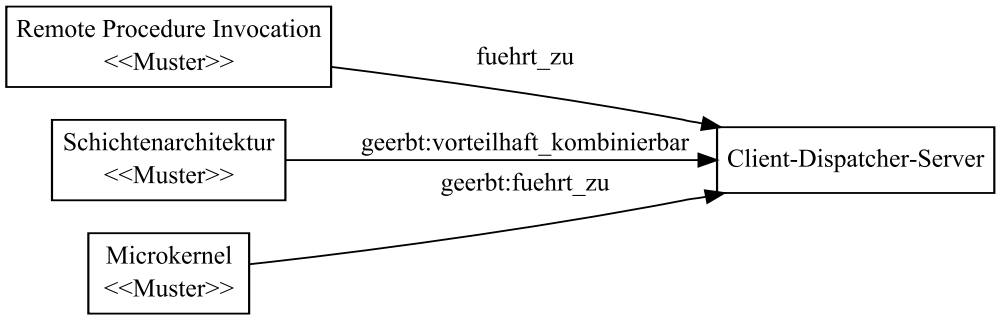
Gruppe(n)

- Kopplung und Entkopplung
- Verbergen der Kommunikation

Kurzbeschreibung

Das Pattern "Client-Dispatcher-Server" führt zwischen Client und Server eine zusätzliche Schicht, die Dispatcher-Schicht ein. Sie dient dazu, den Zugriff vom Client zu dem oder den Servern ortstransparent zu gestalten.

Beziehungen - Grafik




Beziehungen - Detail

- **Remote Procedure Invocation -> Client-Dispatcher-Server** : Ortstransparenz
  Die Ortstransparenz ist die zentrale Zielsetzung von Client-Dispatcher-Server. Dieser Zusammenhang wird in Pattern Oriented Software Architecture, Volume 1, S. 324, Abschnitt Solution beschrieben.
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und

Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.

- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *37. Client-Server*

Quellen

**Taschenbuch der Informatik**

S. 421,

Die Quelle beschreibt die Client-Server-Architektur allgemein und nicht in Form einer Pattern-Schablone. Es beschreibt eine Architektur, in der die Kommunikanten eindeutig den Rollen Client und Server zugeordnet sind. Das Client-Server-Architekturkonzept lässt sich in Form eines Pattern beschreiben. Auf diese Beschreibung wird in dieser Arbeit allerdings aufgrund der allgemeinen Bekanntheit des zugrunde liegenden Konzepts verzichtet.

Taschenbuch der Informatik, Uwe Schneider und Dieter Werner, München: Fachbuchverl. Leipzig im Carl-Hanser-Verl., 2007

Schwerpunkt(e)

- Kommunikation

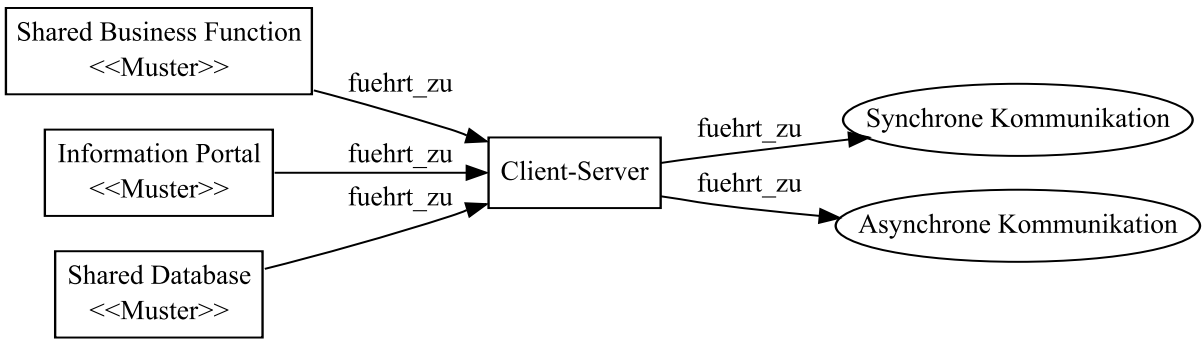
Gruppe(n)

- Rolle der Kommunikanten

Kurzbeschreibung

Das Pattern "Client-Server" beschreibt eine aus zwei verteilten Schichten bestehende Architektur, bei der Kommunikation zwischen einem Bestandteil "Dienste-Anbieter" und einem Bestandteil "Dienste-Konsument" stattfindet.

Beziehungen - Grafik



Beziehungen - Detail

- **Shared Business Function -> Client-Server** : besitzt eindeutige Anbieter/Konsument Beziehung
  Das Shared-Business-Function-Pattern beschreibt sehr allgemein die gemeinsame Nutzung von fachlichen Funktionen in verschiedenen Anwendungen eines Unternehmens oder im vorliegenden Fall in medizinischen Versorgungseinrichtungen. Bietet ein System anderen Systemen eine Teilmenge seiner Funktionen zur gemeinsamen Nutzung an, so tritt dieses System eindeutig in der Rolle des Servers auf, während alle Systeme, die diese Funktionen nutzen die Rolle von Clients einnehmen.
- **Information Portal -> Client-Server** : eindeutige Rollenzuordnung
  Das Information Portal-Pattern beschreibt eine Systemarchitektur, in der der Zugriff auf eine Menge von Backend-Systemen (in der Rolle Server) über einen zentralen Zugriffspunkt (Information Portal) kanalisiert wird. In einer solchen Architektur bleibt das eindeutige Rollenverhältnis zwischen Client und Server erhalten.
- **Shared Database -> Client-Server** :
  Systeme, die **direkt** auf der Verwendung einer gemeinsamen Datenbank aufbauen sind Client-Server-Architekturen, da die Datenbank die Rolle eines gemeinsamen Servers einnimmt und in keiner zu erwartenden Kommunikationssituation auch als Client aktiv wird.
- **Client-Server -> Synchrone Kommunikation** : realisierbar mit
  Client-Server-Systeme sind mit Mitteln synchroner Kommunikation realisierbar
- **Client-Server -> Asynchrone Kommunikation** : realisierbar mit
  Client-Server-Systeme sind mit Mitteln asynchroner Kommunikation realisierbar

## *38. Code based Transformation*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Datentransformation

Kontext

Ein Systembestandteil soll z. B. Dokumente verschiedener Standards oder verschiedener Formate, die den gleichen Inhaltstyp besitzen lesen und auf eine gemeinsame Art weiterverarbeiten können.

Problem
Unterschiedliche Formate lassen sich nicht direkt in die gleichen lokalen Datenstrukturen einlesen. Die Umsetzung einer generischen Routine zur Weiterverarbeitung ist deutlich aufwendiger als die einer Routine, die auf einem konkreten Datentyp aufsetzen kann. Folglich

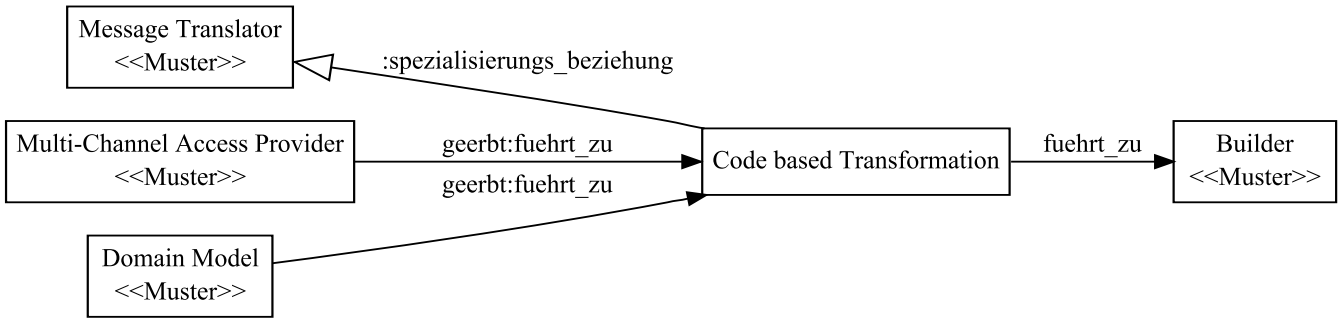
müssen die eingelesenen Daten in einen gemeinsamen, zur Weiterverarbeitung geeigneten Typ umgewandelt werden.

Lösung

Werden in dem zu entwickelnden System nur wenige Transformationen (Umwandlungen in einen anderen Typ) benötigt, muss die Transformation in möglichst kurzer Zeit erfolgen oder ist eine Veränderung der Typen nicht zu erwarten, so ist die Umsetzung der Transformation in Quellcode einer Programmiersprache sinnvoll. Üblicherweise wird dazu die Sprache, in der das System selbst entwickelt ist oder eine darin integrierbare Skriptsprache verwendet.

Beispiel

Beziehungen - Grafik




Beziehungen - Detail

- **Message Translator -> Code based Transformation** :
  Die Code based Transformation ist eine spezielle Ausprägung des Message-Translator-Patterns.
- **Code based Transformation -> Builder** : Umsetzung unterschiedl. Transformationen
  Zur Entkopplung des Transformationsvorgangs von der eigentlichen Anwendungs- und Übertragungslogik kann die Code based Transformation durch die Anwendung des Builder-Patterns gekapselt werden. Dadurch wird es auch möglich, verschiedene Transformationen durch den gleichen steuernden Code (Director) durchzuführen.
- **Multi-Channel Access Provider -> Datentransformation** : Umsetzung von Nachrichten in unterschiedliche Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zur Umsetzung des Multi-Channel-Access-Provider-Pattern wird die Verwendung von mindestens einem Pattern der Gruppe Datentransformation benötigt, um die verschiedenen Zugriffsarten durch jeweils geeignet umgesetzte Nachrichten zu unterstützen.
- **Domain Model -> Datentransformation** : Wenn: Rich Domain Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 117, Patterns of Enterprise Application Architecture, Folwer
  *"A rich Domain Model is better for more complex logic, but is harder to map to the database. A simple Domain Model can* [...]*, whereas a rich Domain Model requires Data Mapper."*

## *39. Command*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 233 ff.

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

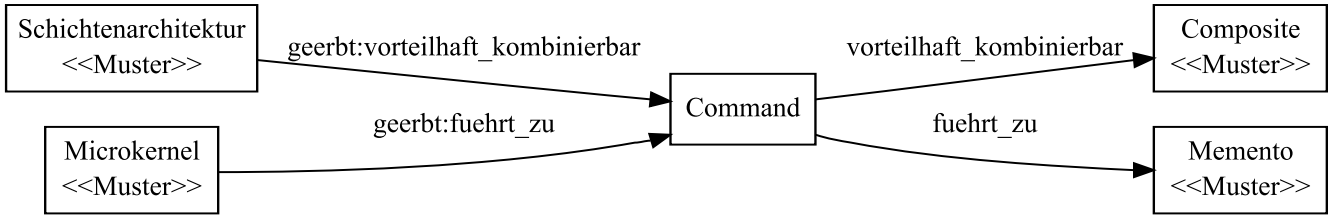
Gruppe(n)

- Kopplung und Entkopplung

Kurzbeschreibung

Das Command-Pattern beschreibt den Ansatz, Operationen in konkrete Implementierungen einer abstrakten Klasse oder eines Interfaces Command abzubilden, sodass die Operation - einschließlich ihres evtl. relevanten Kontexts - wie ein normales Objekt z. B. als Parameter an den Ort ihrer Ausführung übergeben werden kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Command -> Composite** : Zum Aufbau kombinierter Commands
  Siehe Gamma, letzte Seite, Design Pattern Relationships;
  Details siehe S. 237; Abschnitt Consequences; Punkt 3
- **Command -> Memento** : Erreichung von Idempotenz
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships;
  Quelle: Design Patterns; Gamma et al.; S. 239; Punkt 3.
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen

Geerbte Beziehung: fuehrt_zu_beziehung
Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *40. Complete Parameter Checking*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 94 ff.

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

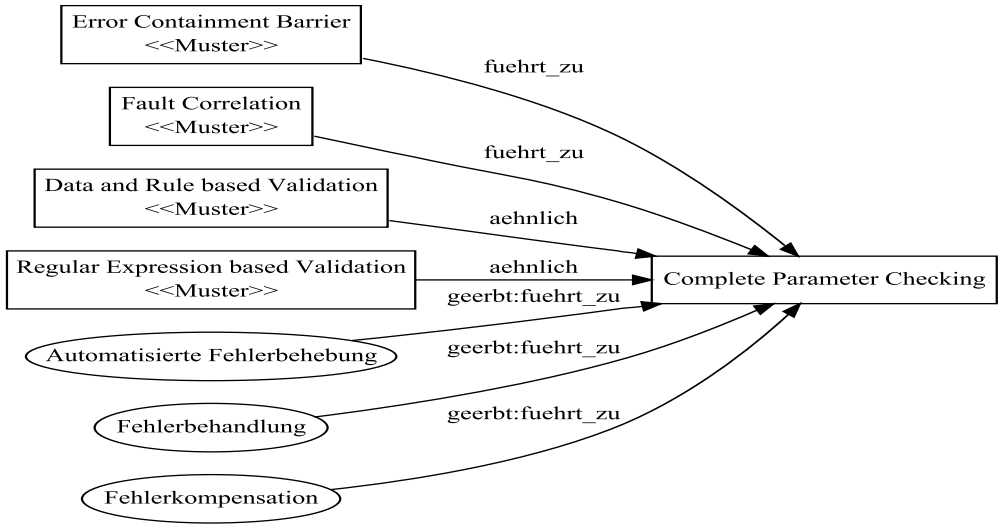
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Complete Parameter Checking" beschreibt ein Verfahren der Fehlervermeidung durch die Prüfung der Parameter vor ihrer eigentlichen Verarbeitung.

Beziehungen - Grafik




Beziehungen - Detail

- **Error Containment Barrier -> Complete Parameter Checking** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87, Abbildung 28;

- **Fault Correlation -> Complete Parameter Checking** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *41. Comply-with-Standard Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 71

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation

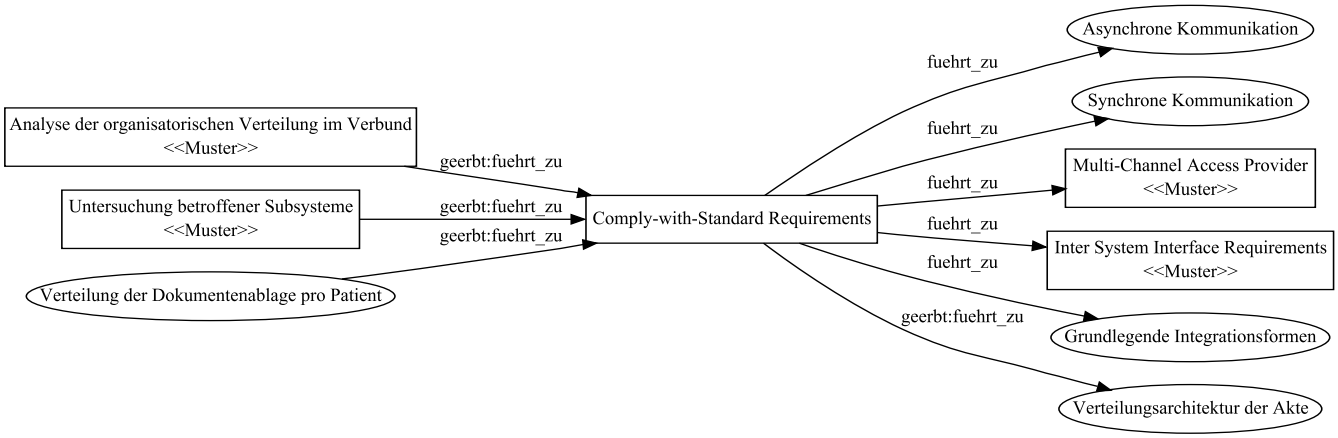
Gruppe(n)

- Kommunikationsanforderungen

Kurzbeschreibung

Das Pattern "Comply-with-Standard Requirements" beschreibt die Rahmenbedingungen und eine Beschreibungsform für die Definition von Anforderungen, die beschreiben, zu welchen Standards usw. ein bestimmtes Softwaresystem kompatibel sein muss.

Beziehungen - Grafik




Beziehungen - Detail

- **Comply-with-Standard Requirements -> Asynchrone Kommunikation** : wenn im Standard gefordert
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Comply-with-Standard Requirements -> Synchrone Kommunikation** : wenn im Standard gefordert
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Comply-with-Standard Requirements -> Multi-Channel Access Provider** : wenn synchrone und asynchrone Standards in Anforderungen
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen. Wenn eine Vielzahl von Standards gefordert wird, dann eignet sich ggf. die Anwendung eines Musters zur Bewältigung dieser Vielfältigkeit.
- **Comply-with-Standard Requirements -> Inter System Interface Requirements** : beeinflusst
  Quelle: Siehe S. 51, Punkt Related patterns; in Software Requirement Patterns; Withall
  Die Beschreibung zum Inter-System-Interface-Requirement-Pattern beschreibt das Comply-with-Standard-Requirement-Pattern als abhängiges Pattern.
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung

Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.

- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** :
  Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *42. Component Configurator*

Quellen

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 490-491

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Flexibilität

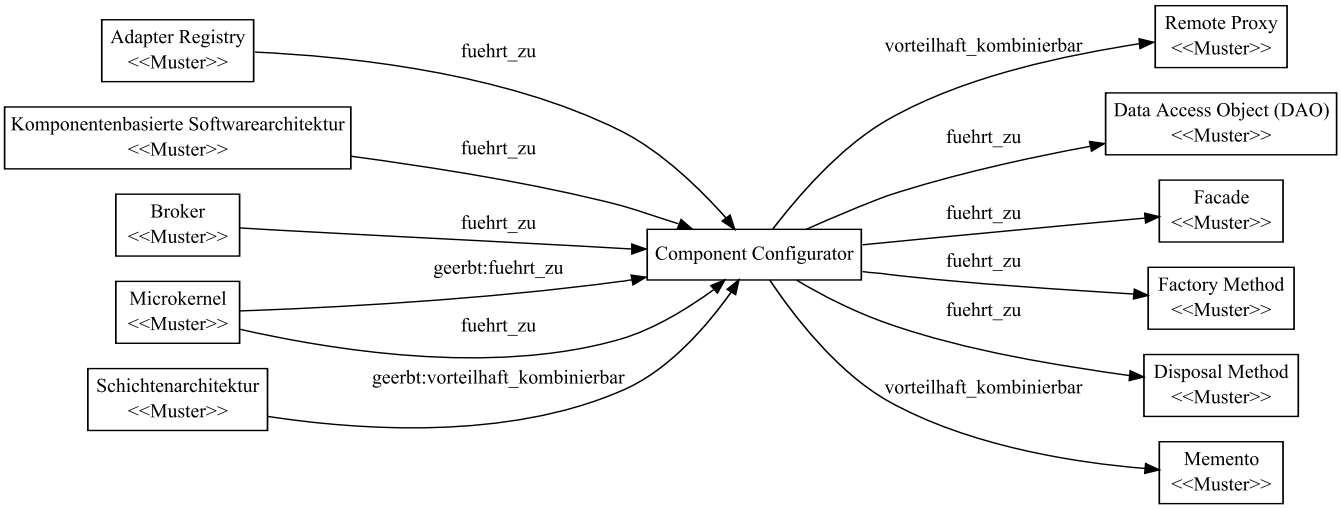
Gruppe(n)

- Kopplung und Entkopplung

Kurzbeschreibung

Das Component-Configurator-Pattern beschreibt einen Weg, Komponenten mittels Konfiguration dynamisch zur Laufzeit in ein System einzubinden.

Beziehungen - Grafik




Beziehungen - Detail

- **Adapter Registry -> Component Configurator** : Konfigurierbarkeit von Adaptoren und Registry
  Sowohl die Adaptoren als auch die Registry werden als per Konfiguration austauschbare Komponenten umgesetzt.
- **Komponentenbasierte Softwarearchitektur -> Component Configurator** : benötigt als Bestandteil
  Um eine komponentenbasierte Architektur umsetzen zu können, in der Komponenten dynamisch konfigurierbar sind, wird zur Umsetzung das Component-Configurator-Pattern benötigt.
- **Broker -> Component Configurator** : Komponentenkonfiguration zur Laufzeit
  **Quelle:** S. 490; Pattern oriented Software Architecture, Volume 4
  Bei der Realisierung einer Broker-Middleware, in der zur Laufzeit flexible Komponentenkonfiguration möglich sein soll, ist die Anwendung des Component-Configurator-Patterns folgerichtig.
  **Quelle:** S. 239; Pattern oriented Software Architecture, Volume 4
  Ein Component Configurator kann in Verbindung mit dem Broker-Muster verwendet werden, um die Requestor und Invoker Implementierung konfigurierbar (z. B. per Konfiguration austauschbar) zu gestalten.
- **Microkernel -> Component Configurator** : Komponentenkonfiguration zur Laufzeit
  Quelle: S. 490; Pattern oriented Software Architecture, Volume 4
- **Component Configurator -> Remote Proxy** : Anbindung entfernter Komponenten
  Zur konfigurierbaren Anbindung entfernter Komponenten (z. B. von Webservices etc.) kann das Component-Configurator-Pattern vorteilhaft mit dem Remote-Proxy-Pattern kombiniert werden.
- **Component Configurator -> Data Access Object (DAO)** : Verwaltung der Konfigurationsdaten
  Die konkrete Speicherung der Konfigurationsdaten der Komponenten kann durch die Anwendung des Data-Access-Object-Pattern verborgen werden. Dadurch wird die eigentliche Component-Configurator-Implementierung von Persistierungscode befreit.

- **Component Configurator -> Facade** : Verbergen der Komponentenschnittstelle
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Component Configurator -> Factory Method** : Laden einer Komponente
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Component Configurator -> Disposal Method** : Entladen einer Komponente
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Component Configurator -> Memento** :
  Quelle: POSA 4; Buschmann et al.; S. 414
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *43. Composite*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 163 ff.

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

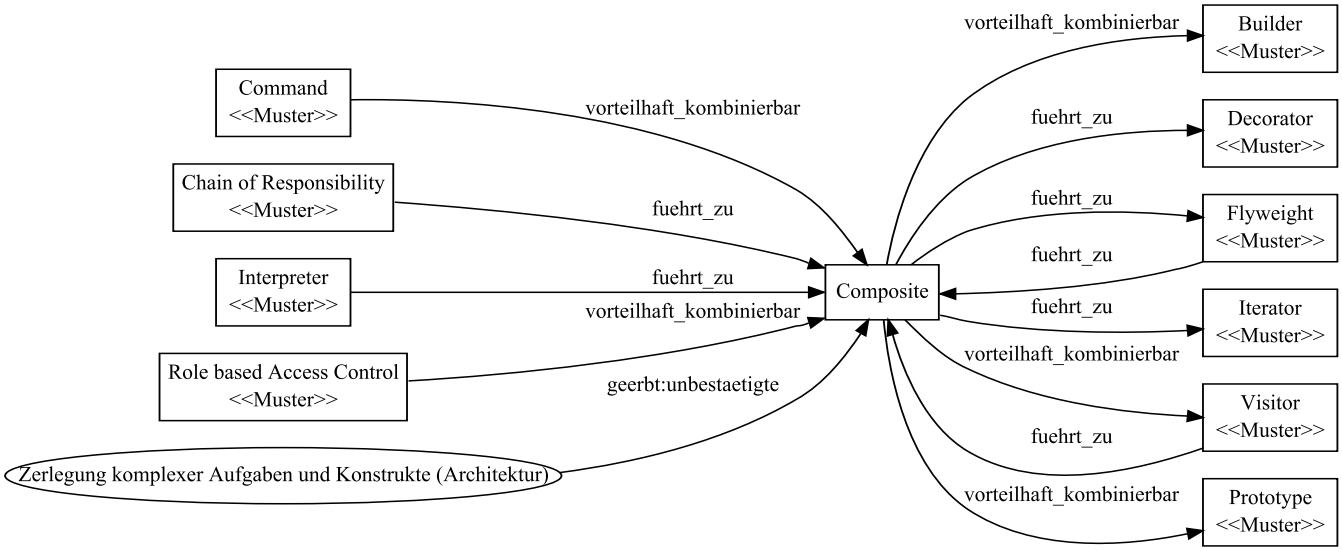
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

Kurzbeschreibung

Das Pattern "Composite" beschreibt die Abbildung von Baum-Strukturen in Form von hierarchisch angeordneten Eltern-Kind-Beziehungen zwischen Objekten.

Beziehungen - Grafik




Beziehungen - Detail

- **Command -> Composite** : Zum Aufbau kombinierter Commands
  Siehe Gamma, letzte Seite, Design Pattern Relationships;
  Details siehe S. 237; Abschnitt Consequences; Punkt 3
- **Chain of Responsibility -> Composite** : wenn existierende Links verwendet werden
  Siehe Design Patterns, Gamma et al. S. 232:
  Das Chain-of-Responsibility-Pattern wird häufig in Verbindung mit dem Composite-Pattern verwendet. Dabei kann z. B. die Eltern-Komponente als Successor einer Komponente verwendet werden.
  S. 227 der gleichen Quelle beschreibt das als die Ausprägung des Chain-of-Responsibility-Patterns, die existierende Beziehungen zwischen den Objekten verwendet. Davon abweichend existiert auch eine Ausprägung, in der speziell für die Anwendung des Chain-of-Responsibility-Patterns eigene Beziehungen eingebaut werden.
- **Interpreter -> Composite** : defining grammar
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Flyweight -> Composite** :
  Quelle: Design Patterns; Gamma et al.; S. 206
  *"The Flywight pattern is often combined with the Composite pattern to implement a logically hierarchical structure in terms of a directed-acyclic graph with shared leaf nodes."*
- **Role based Access Control -> Composite** : Composit Roles
  Quelle: Security Patterns; Schumacher et al.; S. 251
  *"The model shown in the figure on page 252 additionally considers composite roles - it is an application of Composite - and separation of administration from other rights, an application of the policy of separation of duties."*
- **Visitor -> Composite** : Bestandteil der Umsetzung
  Quelle: Design Patterns; Gamma et al; S. 334-335

Das Visitor Pattern hat die Bestandteile Visitor; ConcreteVisitor; Element; ConcreteElement und ObjectStructure. Der Bestandteil ObjectStructure hat folgende Eigenschaft: *"ObjectStructure* [...] *may either be a composite or a collection such as a list or a set."*

- **Composite -> Builder** : createing composites
  Quelle: Design Patterns, Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma et al.; S. 106
- **Composite -> Decorator** : adding responsibilities to objects
  Quelle: Letzte Seite; Design Pattern Relationships; Design Patterns; Gamma et al.
- **Composite -> Flyweight** : sharing composites
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Composite -> Iterator** : enumerating children
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Composite -> Visitor** : Zusaetzliche Operationen hinzufuegen
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Composite -> Prototype** :
  Quelle: Design Patterns; Gamma et al.; S. 126;
  *"Designs that make heavy use of Composite and Decorator patterns often can benefit from Prototype as well."*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *44. Concentrated Recovery*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 145-147

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

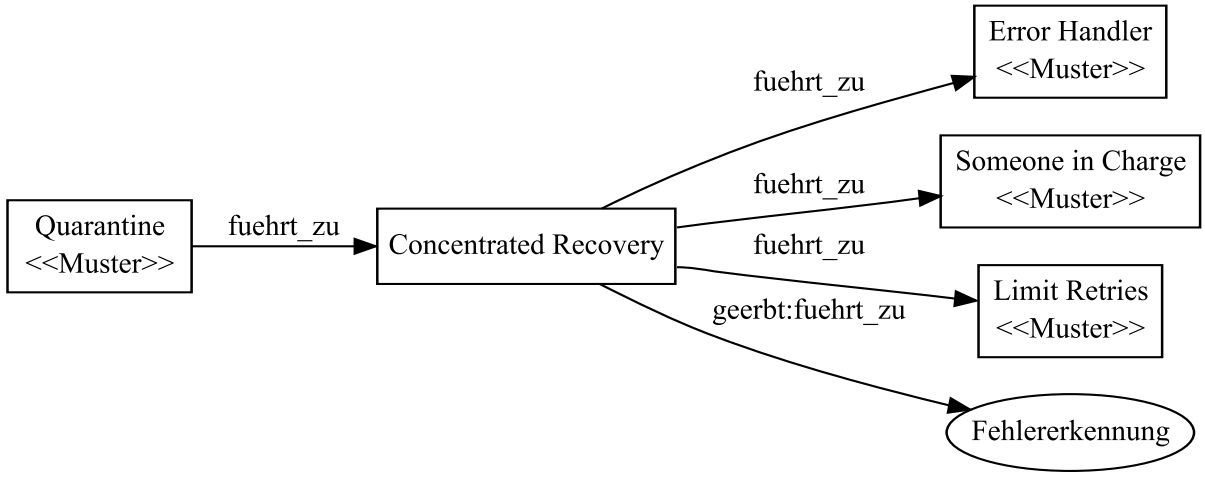
Gruppe(n)

- Automatisierte Fehlerbehebung

### Kurzbeschreibung

Das Pattern "Concentrated Recovery" beschreibt den Weg, die Systemverfügbarkeit durch möglichst schnelle Recovery-Prozesse möglichst groß zu halten. Dazu empfiehlt das Pattern, Recovery-Prozessen maximale Priorität einzuräumen, sodass diese schnellstmöglich abgeschlossen sind.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Quarantine -> Concentrated Recovery** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Concentrated Recovery -> Error Handler** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Concentrated Recovery -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Concentrated Recovery -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## 45. Correcting Audits

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 42-46

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

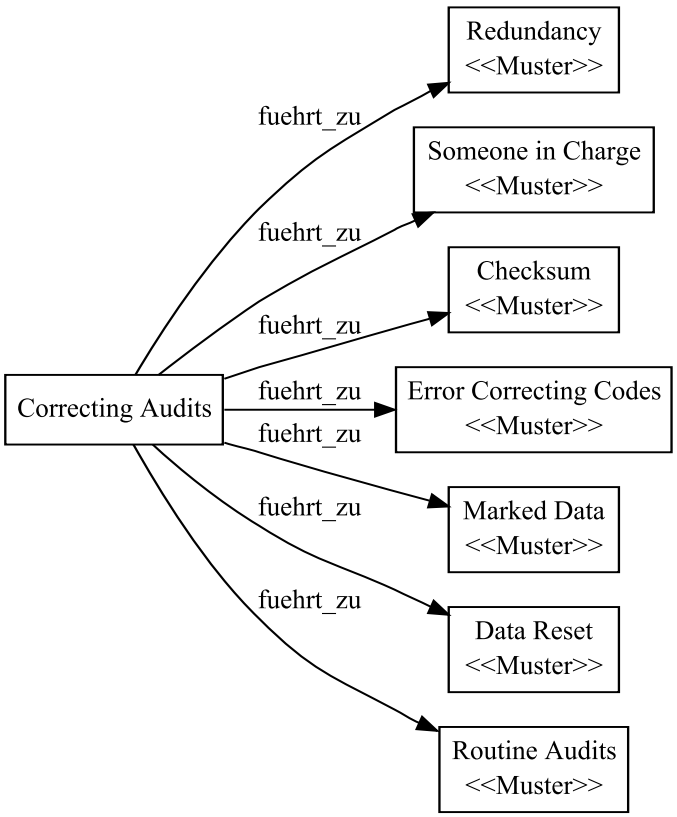
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Correcting Audits" beschreibt die Zweckmäßigkeit einer regelmäßigen oder systematischen Prüfung der Gültigkeit der Daten eines Systems auf der Basis verschiedener möglicher Kriterien.

Beziehungen - Grafik



Beziehungen - Detail

- **Correcting Audits -> Redundancy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Correcting Audits -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Correcting Audits -> Checksum** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Correcting Audits -> Error Correcting Codes** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Correcting Audits -> Marked Data** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Correcting Audits -> Data Reset** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Correcting Audits -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *46. Credentials*

Quellen

**Paper: Credential; P. Morrison and E.B.Fernandez**

P. Morrison and E.B.Fernandez, "Credential", Pattern Languages of Programs Conference (PLoP 2006), http://hillside.net/plop/2006/Papers/Library/PLoP2006_Credential.pdf

Schwerpunkt(e)

- Sicherheit

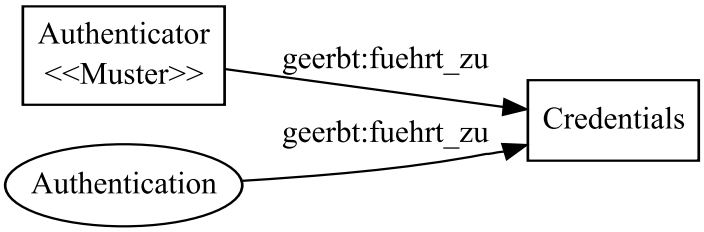
Gruppe(n)

- Authentication Data

Kurzbeschreibung

Das Credentials-Pattern bietet eine Lösung für die Speicherung von Authentifizierungs- und Autorisierungsinformationen in verteilten Systemen.

Beziehungen - Grafik




Beziehungen - Detail

- **Authenticator -> Authentication Data** : Speicherung bzw. Verwaltung der Authentifizierungsdaten
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zur Authentifizierung/Authentisiserung werden Daten benötigt, die ein Subjekt eindeutig identifizieren und solche, die es dem Subjekt möglich machen, nachzuweisen, ein bestimmtes Subjekt zu sein. Diese Daten zu Subjekten müssen verwaltet werden.
- **Authentication -> Authentication Data** : als Datenquelle
  Geerbte Beziehung: fuehrt_zu_beziehung
  Für die Authentifizierung von Benutzern oder Geräten werden Authentifizierungsdaten z. B. in Form von Credentials benötigt.

## *47. Data Access Object (DAO)*

Quellen

**Core J2EE Patterns**

Core J2EE Patterns: Best Practices and Design Strategies, Dan Malks, Deepak Alur, und John Crupi, 1. Aufl. (Pearson Education, 2001).

***Alternative Online-Quelle:*** Core J2EE Pattern Catalog; Core J2EE Patterns - Data Access Object; http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html

Schwerpunkt(e)

- Zuverlässigkeit
- Flexibilität

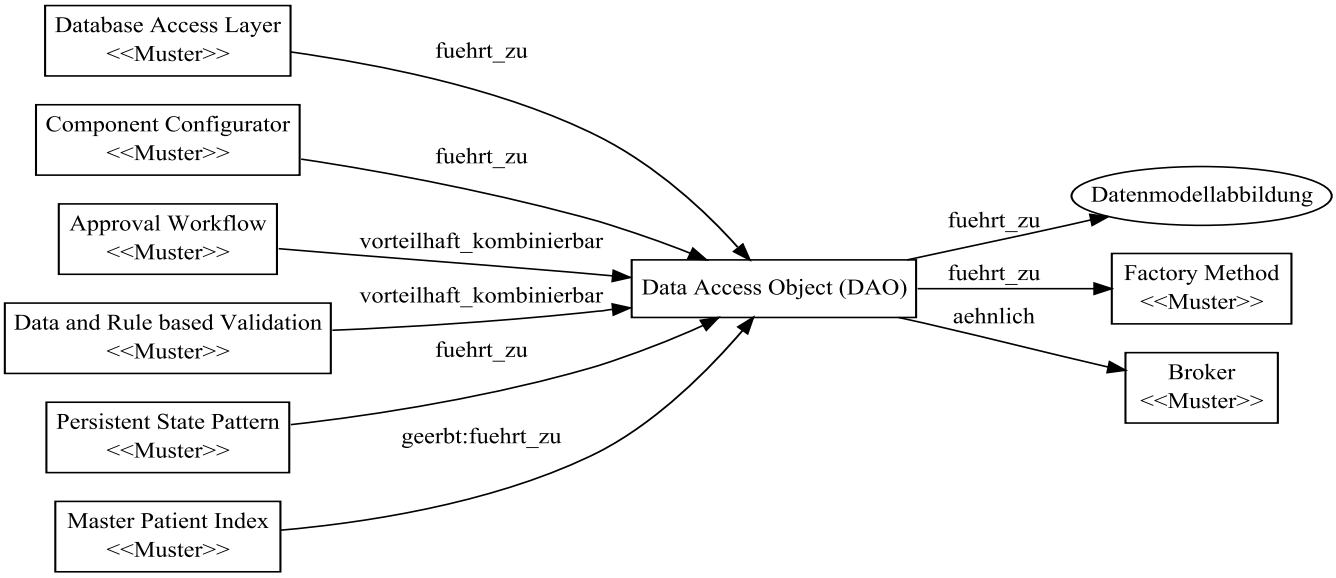
Gruppe(n)

- Verbindung zwischen Anwendungslogik und Daten

Kurzbeschreibung

Das DAO-Pattern beschreibt einen Weg zur objektorientierten Kapselung des Zugriffs auf Daten, die in Datenbanken (oder ggf. auch auf Dateisystemen) abgelegt sind.

Beziehungen - Grafik




Beziehungen - Detail

- **Database Access Layer -> Data Access Object (DAO)** : umsetzbar durch
  Durch die Anwendung des DAO-Patterns kann eine, im Rahmen der Architekturplanung vorgesehene Datenabstraktionsschicht umgesetzt werden. Das DAO-Pattern liefert die Ergebnisse des Zugriffs auf einen definierten Datenspeicher in Form von Objekten an den Aufrufer zurück. Diese Objekte müssen nicht zwangsläufig 1:1 Repräsentationen der physikalischen Speicherungsform sein. Dadurch kann mittels DAO eine Abstraktion von der physikalischen Speicherung erreicht werden.
- **Component Configurator -> Data Access Object (DAO)** : Verwaltung der Konfigurationsdaten
  Die konkrete Speicherung der Konfigurationsdaten der Komponenten kann durch die Anwendung des Data-Access-Object-Pattern verborgen werden. Dadurch wird die eigentliche Component-Configurator-Implementierung von Persistierungscode befreit.
- **Approval Workflow -> Data Access Object (DAO)** : Speicherung der Konfigurationsdaten
  Um den Zugriff auf die Konfigurationsdaten (Regeln) zu vereinfachen ist eine Kombination mit dem DAO-Pattern sinnvoll.
- **Kandidat: Data and Rule based Validation -> Data Access Object (DAO)** : Ermittlung der Quellmenge
  Für die Validierung eines Wertes auf Basis von bestehenden Daten wird zur Anwendung der Regel eine Quellmenge gültiger Werte benötigt. Das DAO-Pattern kann verwendet werden, um den regelbasierten Zugriff auf diese Quelldaten zu kapseln.
- **Persistent State Pattern -> Data Access Object (DAO)** : Bestandteil
  Quelle: Quelle: http://www.hillside.net/plop/2010/papers/saude.pdf Nr. 4.5 Structure; *"Data Access Object (DAO): The DAO is an abstraction layer for the communication with the database"*

- **Data Access Object (DAO) -> Datenmodell-Abbildung** : Verwendet eine Ausprägung von
  Das DAO-Pattern dient der Vereinfachung des Zugriffs auf Datenbestände. Diese Datenbestände selbst können entsprechend der Prinzipien eines der Patterns der Gruppe Datenmodell-Abbildung aufgebaut sein.
- **Data Access Object (DAO) -> Factory Method** : Zur Umsetzung einer DAO Factory
  Quelle:
  http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html;
  Abschnitt "Related Patterns";
- **Data Access Object (DAO) -> Broker** :
  Quelle:
  http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html;
  Abschnitt "Related Patterns";
  *"The DAO pattern is related to the Broker pattern, which describes approaches for decoupling clients and servers in distributed systems. The DAO pattern more specifically applies this pattern to decouple the resource tier from clients in another tier, such as the business or presentation tier."*
- **Master Patient Index -> Verbindung zwischen Anwendungslogik und Daten** :
  Speicherung des Index
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Daten des Master Patient Index müssen persistent gepeichert werden um dauerhaft verwendet werden zu können.

## *48. Data Replication*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 6-7

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

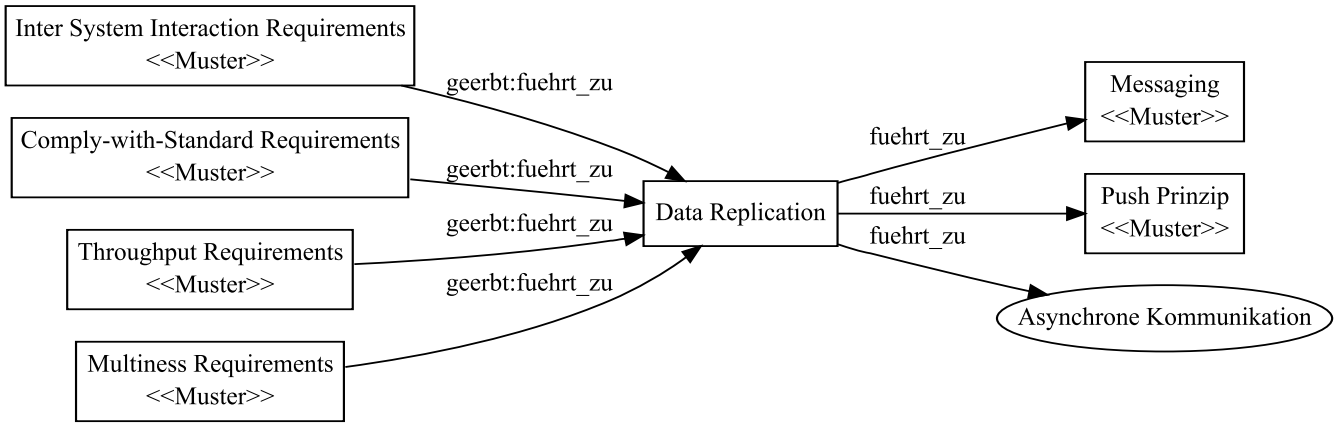
Gruppe(n)

- Grundlegende Integrationsformen

Kurzbeschreibung

Das Pattern "Data Replication" beschreibt den Ansatz, unterschiedliche Systeme durch die Replikation von gemeinsam benötigten Daten zu integrieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Data Replication -> Messaging** : mögl. Umsetzung mit Messaging
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion;
  *"There are many different ways to implement data replication. For example,* [...] *or we can use message-oriented middleware to transport data records inside messages."*
- **Data Replication -> Push Prinzip** : Basiert auf Push
  Änderungen werden zwecks Replikation aktiv an die Datensenken übermittelt. Dieses Vorgehen ist notwendig, da nur das System, in dem ein Datensatz geändert wird, von der Änderung Kenntnis hat. Aktives Nachfragen (Pull) durch die Systeme, die ein Replikat der Änderung erhalten sollen, wäre nur durch Polling (regelmäßiges Nachfragen ohne inhaltlich bedingten Auslöser) möglich.
- **Data Replication -> Asynchrone Kommunikation** : verwendet
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion; "*There are many different ways to implement data replication. For example,* [...]".
  Alle der an dieser Stelle aufgeführten Methoden sind der asynchronen Kommunikation zuzuordnen.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.

- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.
- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## *49. Data Reset*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 177-178

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

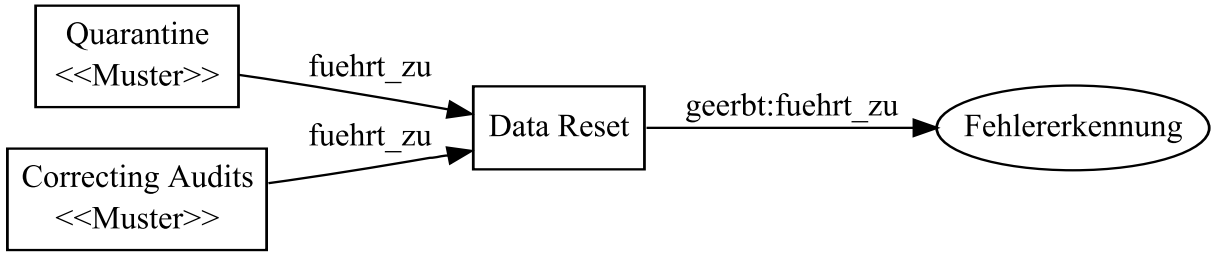
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Data Reset" beschreibt das Verfahren, im Falle eines Fehlers, der Daten des Systems in einen inkonsistenten Zustand gebracht hat, auf einen Ausgangszustand zurückzusetzen. Dieses Verfahren ist z. B. dann nötig, wenn das Pattern "Checkpoint (Error Recovery)" nicht angewendet wurde und somit keine Sicherungspunkte existieren, die für einen Rollback genutzt werden könnten.

Beziehungen - Grafik




Beziehungen - Detail

- **Quarantine -> Data Reset** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141 Abbildung 47
- **Correcting Audits -> Data Reset** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *50. Data and Rule based Validation*

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

- Eingabevalidierung

Kontext

Bei der Eingabe von Daten sollen durch eine sofortige Prüfung eingegebener Werte Fehler verhindert werden. Dabei gibt es unterschiedliche Arten von Werten, die geprüft werden können. Einerseits sind Werte verfügbar bei denen durch bloße Anwendung einer Regel auf den Wert, ohne Zuhilfenahme dritter Daten, deren Gültigkeit festgestellt werden kann. Andererseits gibt es auch Werte, bei denen die Gültigkeit durch den Abgleich mit einer Quellmenge überprüft werden kann. In diesem Fall beschreibt die Regel die Ermittlung und den Vergleich mit der Quellmenge.

Problem
Die Gültigkeit eines Wertes kann nicht allein durch eine Regel und den Wert selbst ermittelt werden.

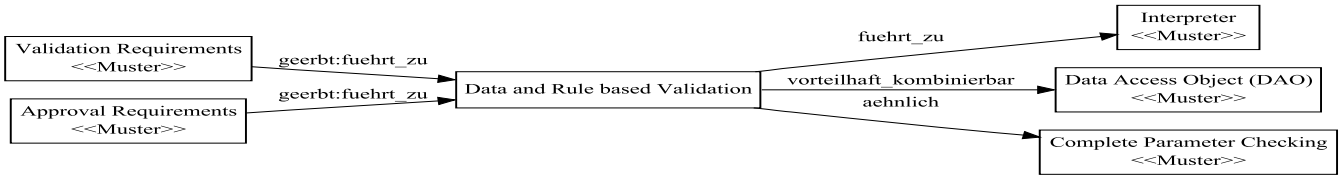
Lösung

Die Validierungsregel beschreibt den Abgleich des Wertes mit einer Menge von Werten. Ist der zu validierende Wert ein Element dieser Menge, so wird der Wert als gültig identifiziert.

Beispiel

- Prüfen einer gültigen Postleitzahl
- Prüfen von bereits bekannten E-Mail-Adressen
- Beliebige Prüfungen bezogen auf Fremdschlüssel-Beziehungen

Beziehungen - Grafik




Beziehungen - Detail

- **Kandidat: Data and Rule based Validation -> Interpreter** : Interpretation der Regeln
  Zur Interpretation der in Strings formulierten Regeln ist ein Interpreter notwendig. Dieser interpretiert die Regeln und wendet sie auf die zu überprüfenden Daten an. Gegebenenfalls werden zur Validierung (z. B. von Fremdschlüsselbeziehungen) aufgrund des Regel-Inhalts auch dritte Daten herangezogen.
- **Kandidat: Data and Rule based Validation -> Data Access Object (DAO)** : Ermittlung der Quellmenge
  Für die Validierung eines Wertes auf Basis von bestehenden Daten wird zur Anwendung der Regel eine Quellmenge gültiger Werte benötigt. Das DAO-Pattern kann verwendet werden, um den regelbasierten Zugriff auf diese Quelldaten zu kapseln.
- **Validation Requirements -> Eingabevalidierung** : Bereich: Dateneingabe
  Geerbte Beziehung: fuehrt_zu_beziehung
  Ein Bereich der Validation Requirements befasst sich mit der Validierung von Daten während deren Eingabe bzw. Übertragung an das System, in dem die Validierung durchgeführt wird.
- **Approval Requirements -> Eingabevalidierung** : Anforderungen für spezielle Eingabevalidierung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Nutzung eines Approval Workflow, also der Bestätigung von eingegebenen Werten durch einen weiteren Benutzer (i.d.R. einen fachlichen Experten), ist eine spezielle Form der Eingabevalidierung.

## *51. Database Access Layer*

Quellen

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 538-539

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit
- Flexibilität

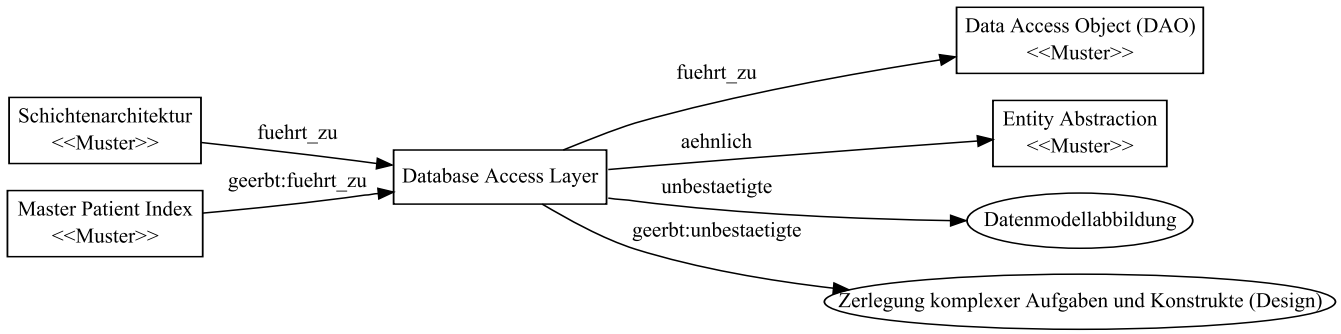
Gruppe(n)

- Verbindung zwischen Anwendungslogik und Daten
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kurzbeschreibung

Das Pattern "Database Access Layer" beschreibt das Konzept, den Bruch zwischen der Datenhaltung in relationalen Datenbanken und der Verwendung der Daten in objektorientiertem Code durch eine eigene Schicht in der Softwarearchitektur zu begegnen. In der Database Access Layer werden die Tabellen der relationalen Datenbanken in Form eines objektorientierten Modells abgebildet.

Beziehungen - Grafik




Beziehungen - Detail

- **Schichtenarchitektur -> Database Access Layer** : Zugriffschicht für Datenquelle(n)
  In einer Schichtenarchitektur kann eine spezielle Schicht für den Zugriff auf ein oder mehrere Datenquellen eingeplant werden. Das ist vor allem dann nützlich, wenn der Zugriff auf diese Datenquelle(n) die Komplexität des Quellcodes so erhöhen würde, dass dieser dadurch weniger übersichtlich wird.
- **Database Access Layer -> Data Access Object (DAO)** : umsetzbar durch
  Durch die Anwendung des DAO-Patterns kann eine, im Rahmen der

Architekturplanung vorgesehene Datenabstraktionsschicht umgesetzt werden. Das DAO-Pattern liefert die Ergebnisse des Zugriffs auf einen definierten Datenspeicher in Form von Objekten an den Aufrufer zurück. Diese Objekte müssen nicht zwangsläufig 1:1 Repräsentationen der physikalischen Speicherungsform sein. Dadurch kann mittels DAO eine Abstraktion von der physikalischen Speicherung erreicht werden.

- **Database Access Layer -> Entity Abstraction** : Hohe Ähnlichkeit
  Das Database-Access-Layer-Pattern weißt Ähnlichkeit mit dem Entity-Abstraction-Pattern aus SOA Design Patterns, Erl, S. 179 auf.
- **Database Access Layer -> Datenmodell-Abbildung** : Abhängig
  Die konkrete Gestaltung der Database Access Layer ist abhängig von der gewählten Form der Datenmodell-Abbildung.
- **Master Patient Index -> Verbindung zwischen Anwendungslogik und Daten** : Speicherung des Index
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Daten des Master Patient Index müssen persistent gepeichert werden um dauerhaft verwendet werden zu können.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *52. Decorator*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 175-184

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

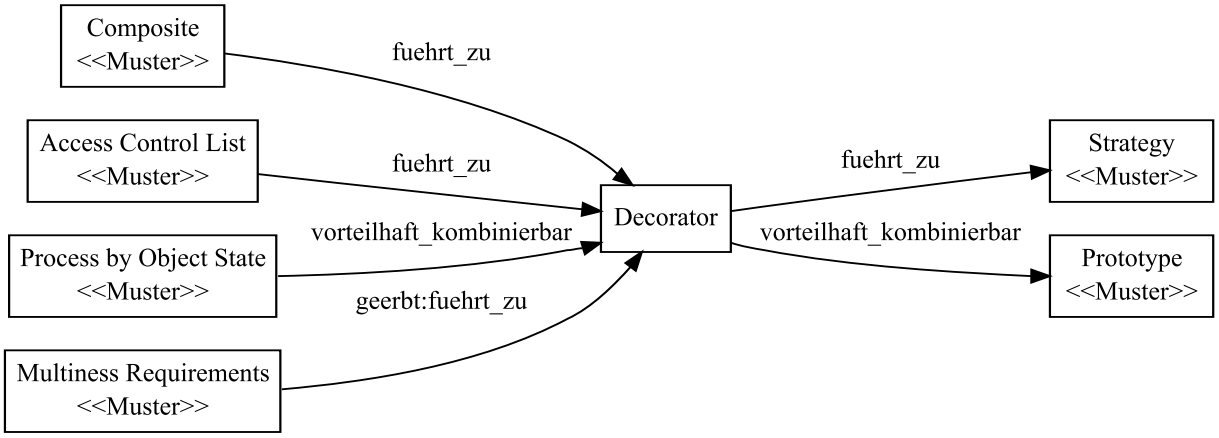
Gruppe(n)

- statische Veraenderung der Schnittstelle
- Handle Body Patterns

## Kurzbeschreibung

Das Pattern "Decorator" beschreibt einen Ansatz, mittels dem einzelne Objekte - und nicht die zugehörigen Klassen - um zusätzliche Eigenschaften oder Funktionen erweitert werden können.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Composite -> Decorator** : adding responsibilities to objects
  Quelle: Letzte Seite; Design Pattern Relationships; Design Patterns; Gamma et al.
- **Access Control List -> Decorator** : can be implemented as
  Die Klassendiagramme in
  http://hillside.net/plop/2007/papers/PLoP2007_DelessyEtAl.pdf Figure 4 sowie Gamma, Johnson, Vlissides, Design Patterns S. 177 legen nahe, dass eine Umsetzung von ACLs mittels Decorator in vielen Fällen sinnvoll ist. Außerdem bestärkt auch die folgende Aussage (aus http://hillside.net/plop/2007/papers/PLoP2007_DelessyEtAl.pdf) diese Vermutung:
  *"Implement the Access Matrix by associating each object with an Access Control List (ACL) that specifies which actions are allowed to be performed on the object and by which authenticated users."*
- **Process by Object State -> Decorator** : Zur flexibleren Gestaltung der Zustandsermittlung
  Wird der Prozesszustand aus den Attributen des Domänenmodells ermittelt, so kann, besonders wenn das System verschiedene Prozesse abbildet, durch die Verwendung des Decorator-Patterns das Domain-Model ohne die zusätzliche Komplexität der Ermittlung des Prozesszustands umgesetzt werden.
- **Decorator -> Strategy** : changing skin versus guts
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Decorator -> Prototype** :
  Quelle: Design Patterns; Gamma et al.; S. 126;
  *“Designs that make heavy use of Composite and Decorator patterns often can benefit from Prototype as well.”*

- **Multiness Requirements -> statische Veraenderung der Schnittstelle** : Indirekt:
  Abbildung auf Ziel-Schnittstellen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Grundsätzlich werden Muster zur statischen Veränderung der Schnittstelle verwendet um einzelne der Multiness Requirements zu erfüllen. Es ist aber zusätzlich ein größeres Pattern (Ebene Architektur) wie z. B. das Multi-Channel-Access-Provider-Pattern notwendig um die Multiness Requirements in einer konsistenten Architektur abzubilden.

## *53. Deferrable Work*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 187-188

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

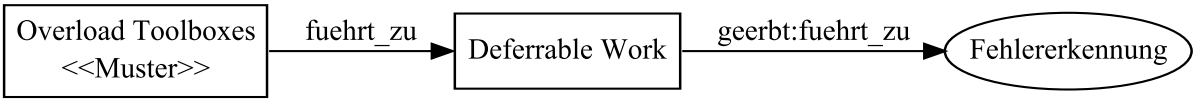
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Deferrable Work" bezieht sich auf Performance-Probleme in Überlast-Szenarios und beschreibt die Rahmenbedingungen unter denen bestimmte Systemaufgaben zugunsten des Erhalts betriebsfähiger Antwortzeiten ausgesetzt werden können.

Beziehungen - Grafik




Beziehungen - Detail

- **Overload Toolboxes -> Deferrable Work** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *54. Demilitarized Zone*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 449-456

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

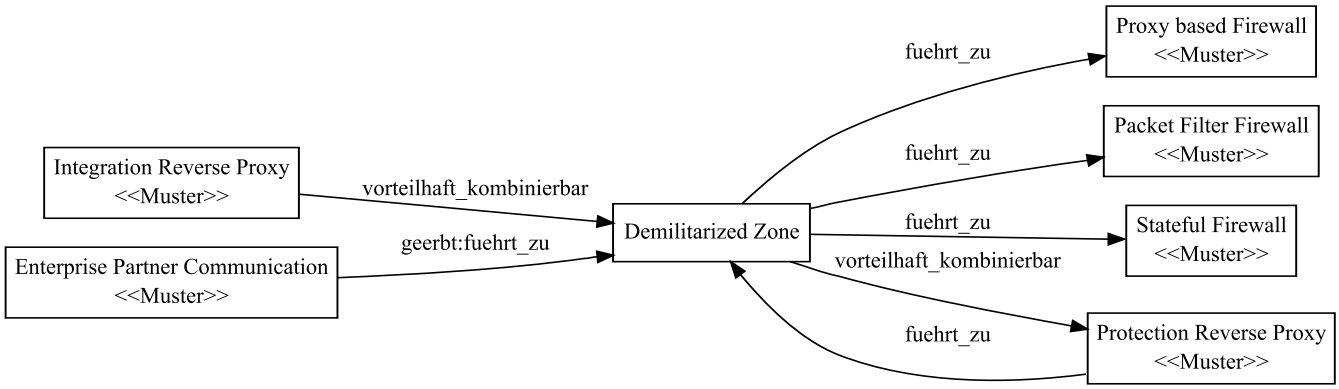
Gruppe(n)

- Access Restriction

Kurzbeschreibung

Das Pattern "Demilitarized Zone" (DMZ) beschreibt das Konzept, Rechner, die aus ungeschützten Netzen erreichbaren Dienste anbieten in einer separaten Zone zu platzieren. Dazu werden sowohl die zu dem ungeschützten Netz als auch die aus dem firmeneigenen Netz bestehenden Verbindungen durch je eine Firewall gefiltert.

Beziehungen - Grafik




Beziehungen - Detail

- **Integration Reverse Proxy -> Demilitarized Zone** : Hilfreiche Rahmenbedingung
  Quelle: S. 467 u. 468; Abbildung „Protection using an Integration Reverse Proxy“; Security Patterns; Schumacher et al.
  Eine Menge von Servern kann durch einen Integration Reverse Proxy in einer Demilitarized Zone (DMZ) gesichert werden.
- **Protection Reverse Proxy -> Demilitarized Zone** :
  Quelle: Security Patterns; Schumacher et al.; S. 459;
  *“*[...] *The resulting network topology provides a Demilitarized Zone containing only the reverse proxy machine, and a secured server zone containing the Web server.”*

- **Demilitarized Zone -> Proxy based Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Demilitarized Zone -> Packet Filter Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Demilitarized Zone -> Stateful Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Demilitarized Zone -> Protection Reverse Proxy** : Anbindung dynamischer Inhalte
  Quelle: S. 451; Security Patterns; Schumacher et al.
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *55. Disposal Method*

Quellen

**Paper: Patterns in Java**

S. 4-5

Patterns in Java: The Importance of Symmetry; K. Henney; JavaSpektrum, Issue 6, 2002
Alternative Online-Quelle:
http://www.two-sdg.demon.co.uk/curbralan/papers/javaspektrum/TheImportanceOfSymmetry.pdf

Schwerpunkt(e)

- Flexibilität

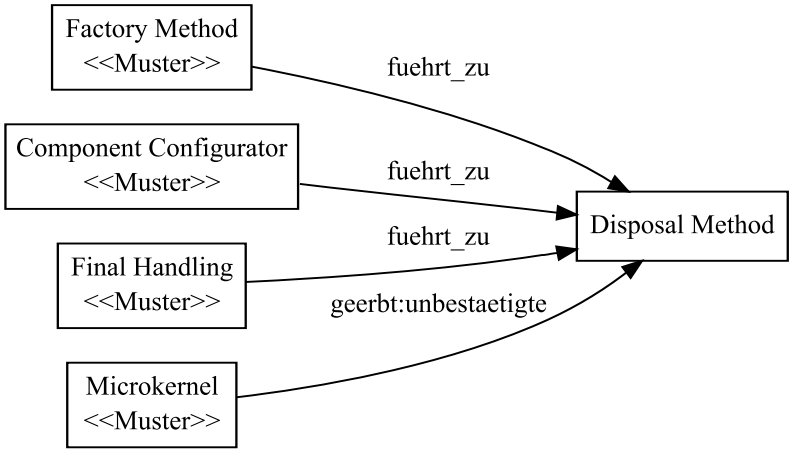
Gruppe(n)

- Instanziierung und Kontrolle der Instanzen

Kurzbeschreibung

Das Pattern "Disposal Method" ist das logische Gegenstück zum Pattern "Factory Method". Es beschreibt das Vorgehen, spezielle Methoden zum Zerstören von nicht mehr benötigten Objekten bereitzustellen. Das kann z. B. dann erforderlich sein, wenn mit dem Zerstören des Objekts ein Abbau von Verbindungen usw. einhergeht.

Beziehungen - Grafik




Beziehungen - Detail

- **Factory Method -> Disposal Method** : dispose created objects
  Quelle: Paper: Patterns in Java; S. 5 unten
- **Component Configurator -> Disposal Method** : Entladen einer Komponente
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Final Handling -> Disposal Method** : Kann durch Disposal Method realisiert werden
  Das Final-Handling-Pattern beschreibt das Freigeben belegter Ressourcen im Fehlerfall. Es kann sowohl in der Ausprägung "Separate Mechanism" als auch in der Ausprägung "Shared Mechanism" durch das Disposal-Method-Pattern realisiert werden.
- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen
  Geerbte Beziehung: unbestaetigte_beziehung
  Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.
  Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

## *56. Distributed Business Process*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 8-9

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

- Flexibilität

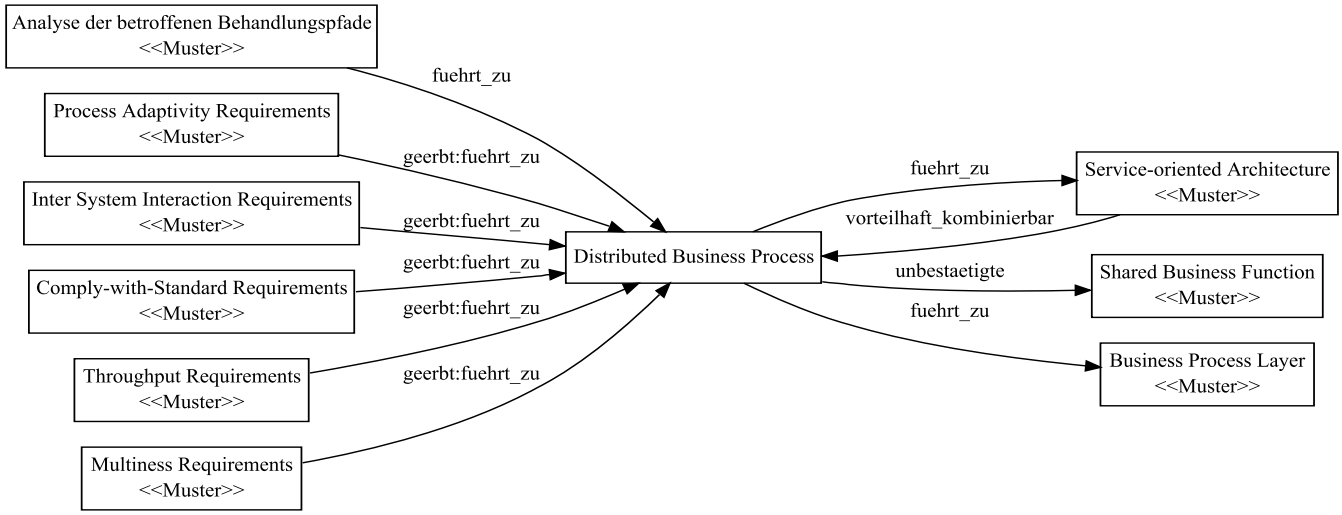
### Gruppe(n)

- Grundlegende Integrationsformen
- Prozessabbildung

### Kurzbeschreibung

Das Pattern "Distributed Business Process" beschreibt allgemein einen Architekturtyp von Anwendungen, die basierend auf verteilten Komponenten oder Subsystemen ein System zur Abbildung eines oder mehrerer Geschäftsprozesse ermöglicht.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Service-oriented Architecture -> Distributed Business Process** : Ergänzen einander sinnvoll
  Quelle: Enterprise Integration Patterns, Hohpe, S. 9 kurz vor Abbildung zu Business-to-Business Integration
  Quelle: SOA in Practice, Josuttis; S. 72; Process-Enabled SOA
  Fazit: Die Prinzipien serviceorientierter Architekturen sind geeignet die Abbildung von verteilten Geschäftsprozessen zu unterstützen.
- **Analyse der betroffenen Behandlungspfade -> Distributed Business Process** :
  Wenn über mehrere Subsysteme verteilt
  Wenn die Analyse der betroffenen Behandlungspfade ergeben hat, dass sich die Prozesse über mehrere beteiligte Subsysteme erstrecken, kann im Verlauf der Architekturkonzeption die Anwendung des Distributed-Business-Process-Patterns vorteilhaft sein.
- **Distributed Business Process -> Service-oriented Architecture** : Benötigt SOA oder Shared Business Functions
  Um einen oder mehrere verteilte Geschäftsprozesse umzusetzen wird als Basis die Existenz von gemeinsam nutzbaren Anwendungsfunktionen (Shared Business

Functions) oder noch umfassender, die Existenz einer Service orientierten Architektur(SOA) benötigt.

- **Distributed Business Process -> Shared Business Function** : Benötigt SOA oder Shared Business Functions
  Um einen oder mehrere verteilte Geschäftsprozesse umzusetzen wird als Basis die Existenz von gemeinsam nutzbaren Anwendungsfunktionen (Shared Business Functions) oder noch umfassender, die Existenz einer Service orientierten Architektur(SOA) benötigt.
- **Distributed Business Process -> Business Process Layer** : Bei Schichtenarchitektur Sollen durch ein Anwendungssystem ein oder mehrere Geschäftsprozesse abgebildet werden, die auf mehr als einem Quellsystem basieren, spricht man von einem verteilten Geschäftsprozess (Distributed Business Process). Sollen verteilte Geschäftsprozesse durch ein gemeinsames Frontend zur Verfügung gestellt werden, das die technisch verteilte Natur des Prozesses gegenüber dem Benutzer transparent macht, so ist zwischen der Benutzeroberfläche und einer Integrationsschicht zur Anbindung der Quellsysteme zusätzlich eine Business Process Layer (siehe POSA4, S. 69 ff.) zweckmäßig.
- **Process Adaptivity Requirements -> Prozessabbildung** : Input: Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Process Adaptivity Requirements sind die Flexibilitätsanforderungen an den gewählten Mechanismus zur Prozessabbildung. Hohe Flexibilitätsanforderungen benötigen umfassende Möglichkeiten zur Konfiguration von Prozessen, niedrige Anforderungen aus relativ statischen Prozessen führen zu stärker direkt im Quellcode des Systems realisierten Prozessen.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.

- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## *57. Document Registry*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Verwaltung verteilter Dokumente

Kontext

Verschiedene Dokumente werden verteilt auf verschiedene Datensenken (Repositories) abgespeichert. Die fachlichen Rahmenbedingungen des Systems machen ein systematisches Auffinden und durchsuchen der verteilt abgespeicherten Dokumente notwendig.

Problem
Dokumente, die verteilt, in einer größeren Zahl verschiedener Repositories abgelegt werden, sollen gezielt aufgelistet und durchsucht werden können. Auch ein Verschieben eines Dokuments von einem Repository in ein anderes soll möglich sein, ohne dadurch das Auffinden des Dokuments zu verhindern.

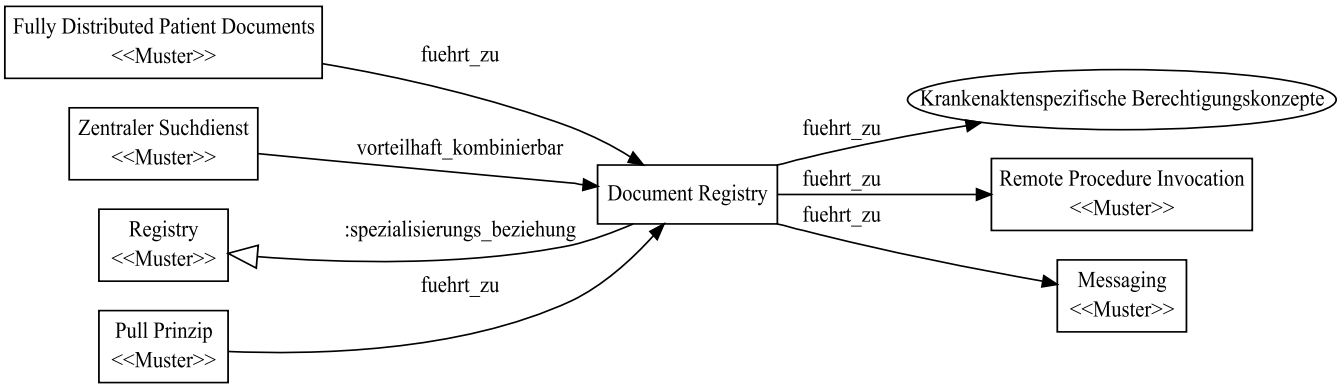
Lösung
Die verschiedenen Datensenken registrieren neu erhaltene Dokumente automatisch bei einer zentralen Komponente, der Document Registry. Sie bietet Operationen zum Registrieren, Auflisten, metadaten-basiertem Suchen und lokalisieren von Dokumenten. Auf diese Weise stellt die Document Registry eine Art gemeinsames Inhaltsverzeichnis für eine Vielzahl verschiedener Repositories dar.

Beispiel
**Beispiel: Virtual shared EHR system**
Quelle: Joachim Bergmann u. a., “An e-consent-based shared EHR system architecture for integrated healthcare networks,” International Journal of Medical Informatics 76, no. 2-3 (Februar): 130-136, doi:10.1016/j.ijmedinf.2006.07.013.; S. 133;
Abbildung 1 und Abbildung 2
**Beispiel: IHE Document Registry**

Quellen:

- http://wiki.ihe.net/index.php?title=Document_Registry
- http://www.ihe.net/Technical_Framework/upload/IHE_ITI_Suppl_XDW_Rev2-0_PC_2011-07-29.pdf

Beziehungen - Grafik




Beziehungen - Detail

- **Fully Distributed Patient Documents -> Document Registry** : zur Registrierung und Wiederfindung
  Wenn Dokumente verteilt im System auf verschiedenen Systemkonten (auch heterogenen Systemknoten) angelegt werden können ist eine zentrale Registrierstelle, an der der Ablageort jedes Dokuments vermerkt ist hilfreich.
- **Zentraler Suchdienst -> Document Registry** : Bei fully distributed Patient Documents
  Wird für die Verteilung der Dokumentenablage das Pattern Fully Distributed Patient Documents gewählt, so eignet sich als Registry für den Suchindex das Document-Registry-Pattern.
- **Registry -> Document Registry** : Ist eine Spezialform von
- **Pull Prinzip -> Document Registry** : Zur Findung abholbarer Dokumente
  Bei der Verwendung des Pull-Prinzips kann eine zentrale Document Registry notwendig (oder zumindest hilfreich) sein, um den Ablageort eines Dokuments zu finden. Die Kenntnis über den Ablageort ist notwendig, um das Dokument aktiv abholen zu können.
- **Document Registry -> Krankenaktenspezifische Berechtigungskonzepte** : Berechtigungsermittlung
  Im Kontext medizinischer Informationssysteme ist häufig bereits das Wissen über die Existenz eines Dokuments zu einem bestimmten Typ von Behandlung datenschutzrechtlich relevant. Aus diesem Grund ist es notwendig auch in einem Verzeichnisdienst, der die Suche über verteilt abgelegte Dokumente ermöglicht, ein umfassendes Berechtigungssystem zu integrieren.
- **Document Registry -> Remote Procedure Invocation** : Aufruf der Registry Methoden
  Um die verschiedenen Methoden der Document Registry von verschiedenen

Systemknoten aus aufrufen zu können ist es notwendig, dass diese Methoden remotefähig zur Verfügung stehen.

- **Document Registry -> Messaging** : Asynchrones registrieren von Dokumenten

## *58. Domain Model*

Quellen

**Patterns of Enterprise Application Architecture, Fowler**

S. 116-124

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

Schwerpunkt(e)

- Flexibilität

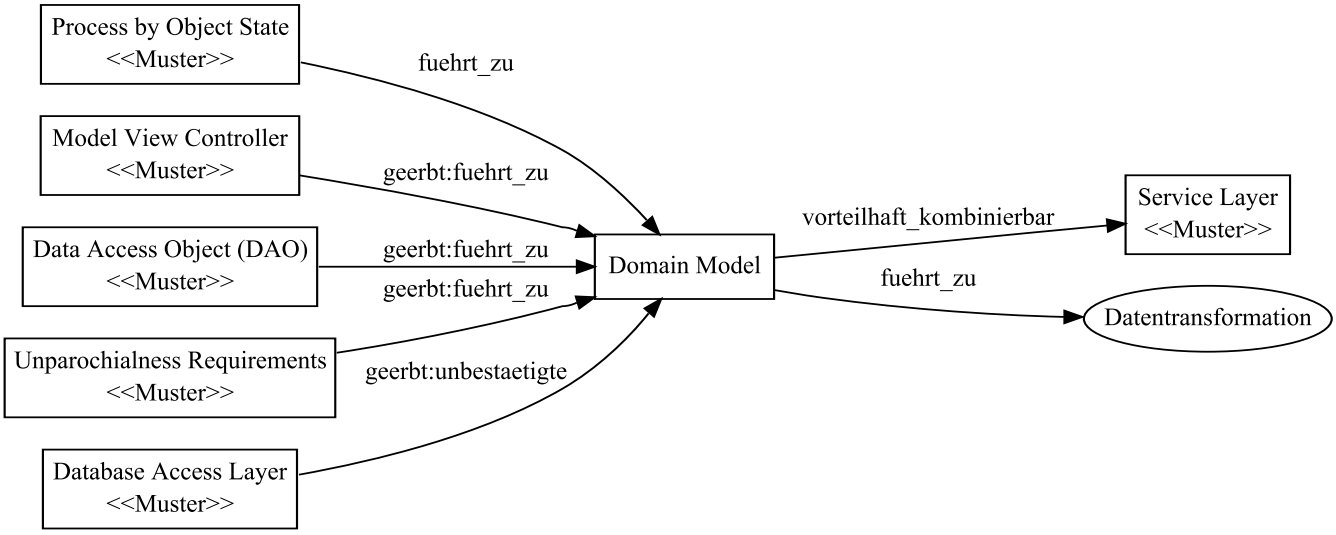
Gruppe(n)

- Datenmodell-Abbildung

Kurzbeschreibung

Das Pattern "Domain Model" beschreibt allgemein die Abbildung der fachlich relevanten Bestandteile der Zieldomäne in Form eines objektorientierten Modells.

Beziehungen - Grafik




Beziehungen - Detail

- **Process by Object State -> Domain Model** : Prozesszustand als Teil des Domänenmodells
  Der Prozesszustand wird als Teil des Domänenmodells direkt in den verschiedenen

Entitäten mit abgebildet. Das kann auch eine Berechnung des Prozesszustands aus den Zuständen verschiedener Attribute bedeuten.

- **Domain Model -> Service Layer** : Konkrete Dienste zur Nutzung des Domain-Model
  Quelle: S. 119, Patterns of Enterprise Application Architecture, Folwer;
  *"Whe you use Domain Model you may want to consider Service Layer to give your Domain Model a more distinct API."*
- **Domain Model -> Datentransformation** : Wenn: Rich Domain Model
  Quelle: S. 117, Patterns of Enterprise Application Architecture, Folwer
  *"A rich Domain Model is better for more complex logic, but is harder to map to the database. A simple Domain Model can* [...]*, whereas a rich Domain Model requires Data Mapper."*
- **Model View Controller -> Datenmodell-Abbildung** : Gestaltung des Bestandteils Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Der Bestandteil "Model" kann in verschiedenen Formen abgebildet werden. Die gängigste Form dabei ist das Domain Model. Das Domain-Model-Pattern ist Bestandteil der Gruppe Datenmodell-Abbildung. Auch die anderen Elemente der Gruppe können zur Umsetzung eines Model im Rahmen der Anwendung des Model-View-Controller-Pattern verwendet werden.
  Quelle: S. 330; Patterns of Enterprise Application Architecture; Fowler;
  *"In its most pure OO form the model is an object within a Domain Model. You might also think of* [...]*"*
- **Data Access Object (DAO) -> Datenmodell-Abbildung** : Verwendet eine Ausprägung von
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das DAO-Pattern dient der Vereinfachung des Zugriffs auf Datenbestände. Diese Datenbestände selbst können entsprechend der Prinzipien eines der Patterns der Gruppe Datenmodell-Abbildung aufgebaut sein.
- **Unparochialness Requirements -> Datenmodell-Abbildung** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Unparochialness-Requirements-Pattern dient der Definition von Anforderungen bezüglich der Fähigkeit eines Systems, sich veränderlichen Umgebungsbedingungen anzupassen. Die verschiedenen Patterns der Gruppe Datenmodell-Abbildung führen zu unterschiedlich flexiblen Datenmodellen. Während z. B. das Entity-Attribute-Value-Pattern zu einem leicht veränderbaren Datenmodell führt, wird beim Domain-Model-Pattern das Datenmodell relativ statisch, dafür aber einfacher verwendbar, abgelegt.
- **Database Access Layer -> Datenmodell-Abbildung** : Abhängig
  Geerbte Beziehung: unbestaetigte_beziehung
  Die konkrete Gestaltung der Database Access Layer ist abhängig von der gewählten Form der Datenmodell-Abbildung.

## *59. Dynamic Capacity Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 212

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Zuverlässigkeit

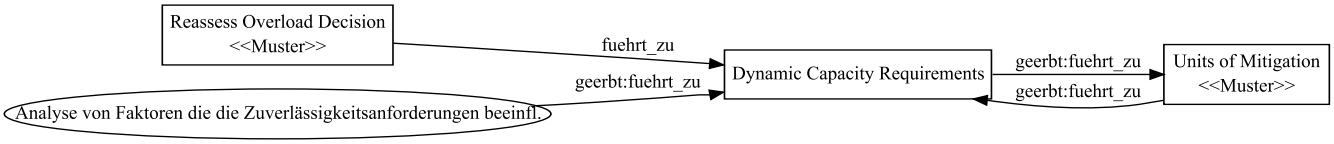
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Dynamic Capacity Requirements" beschreibt ein Modell zur Spezifikation der dynamischen Kapazität eines Systems. Beispiele hierfür sind z. B. die Anzahl der gleichzeitig unterstützten Endbenutzer oder die Anzahl der innerhalb einer definierten Zeiteinheit unterstützten Requests.

Beziehungen - Grafik




Beziehungen - Detail

- **Reassess Overload Decision -> Dynamic Capacity Requirements** : anpassen
  Das Reassess-Overload-Decision-Pattern beschreibt die situationsbedingte Nachbesserung der Mechanismen zur Bewältigung von Überlastungsszenarien. Wird im Rahmen dieses Reassessment-Vorgangs eine dauerhaft höhere Systemlast festgestellt, so führt diese Neubewertung zu einer Anpassung der Dynamic Capacity Requirements.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.

- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.

- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## *60. Einfaktorige Authentifizierung*

Englische Bezeichnung

- One Factor Authentication

Quellen

**IT-Sicherheit, Eckert**

Logisches Gegenteil zu mehrfaktoriger Authentifizierung S. 431 unten

“10 Authentifikation,” in IT-Sicherheit, Bd. 5 (Oldenbourg Wissenschaftsverlag GmbH, 2011), 429-533, http://dx.doi.org/10.1524/9783486595970.429.

Schwerpunkt(e)

- Sicherheit

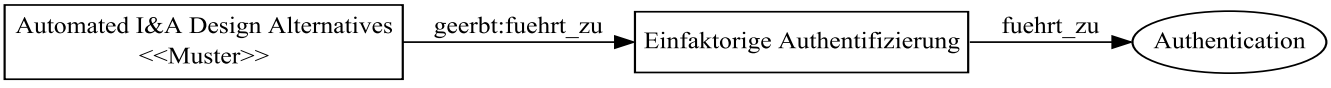
Gruppe(n)

- Anzahl der Authentifizierungskriterien

Kurzbeschreibung

Das Pattern "Einfaktorige Authentifizierung" beschreibt die Formen der Authentifizierung die auf nur einem Faktor (wie z. B. Wissen oder Besitz) basieren.

Beziehungen - Grafik



Beziehungen - Detail

- **Einfaktorige Authentifizierung -> Authentication** : Authentifizierung mit einfachem Credential
  Einfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von nur einem Authentifizierungsverfahren. Typische Beispiele sind:
  - Zugang zum System durch Benutzername/Passwort
  - Zugangssicherung durch Fingerabdruck-Scan

- **Automated I&A Design Alternatives -> Anzahl der Authentifizierungskriterien** :
  Anzahl der Authentifizierungskriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Je nach Sicherheitsbedarf der Applikation muss ein Subjekt ein oder mehrere Credentials richtig befüllen können um sich erfolgreich zu Authentisieren.

## *61. Enterprise Partner Communication*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 173-185

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

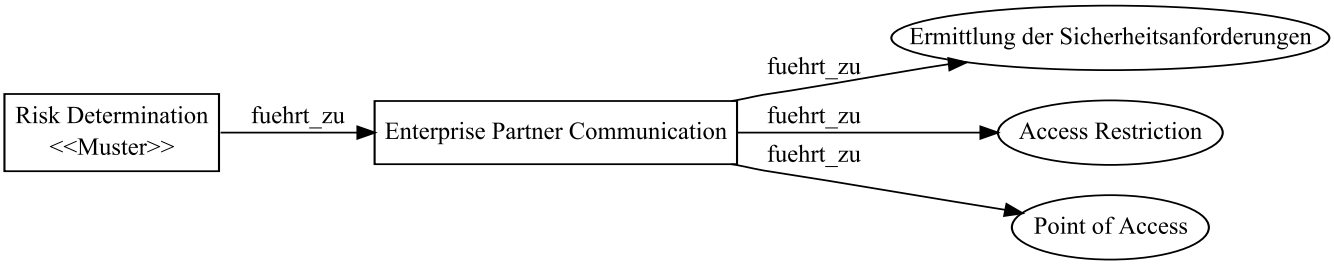
Gruppe(n)

- Analyse von Sicherheitsproblemen und Lösungsansätzen

Kurzbeschreibung

Das Pattern "Enterprise Partner Communication" beschreibt einen Weg, die Eigenschaften der Kommunikationsbeziehungen zu Partner-Unternehmen oder Partner-Einrichtungen so zu dokumentieren oder definieren, dass das Ergebnis für die Analyse der Sicherheitsanforderungen genutzt werden kann.

Beziehungen - Grafik



Beziehungen - Detail

- **Risk Determination -> Enterprise Partner Communication** : Input: Bewertete Risiken
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.
- **Enterprise Partner Communication -> Point of Access** : Gestaltung des Zugriffspunkts für Partner
  Das Enterprise-Partner-Communication-Pattern beschreibt die Kommunkationsanforderungen zwischen Organisationen mit voneinander getrennt existierender IT-Infrastruktur. Zur Umsetzung der Enterprise Partner Communication muss ein Muster für die Gestaltung des Point of Access ausgewählt werden, um für den Kommunikationspartner einen Zugangsweg zu dem betreffenden IT-System zu eröffnen.

## *62. Enterprise Security Approaches*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 148 ff.

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Analyse von Sicherheitsproblemen und Lösungsansätzen

Kurzbeschreibung

Das Pattern "Enterprise Security Approaches" führt durch die Auswahl von geeigneten Arten der Sicherung. Die Arten der Sicherung sind hierbei durch die relativ abstrakten Begriffe Prevention, Detection und Response zusammengefasst.

Beziehungen - Grafik


Beziehungen - Detail

- **Risk Determination -> Enterprise Security Approaches** : Input: Bewertete Risiken
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Enterprise Security Approaches -> Enterprise Security Services** : Input: Lösungsansätze für Risiken
  Beziehung siehe S. 60, 61 in Security Patterns, Schumacher et al.

## *63. Enterprise Security Services*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**
S. 161-172

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

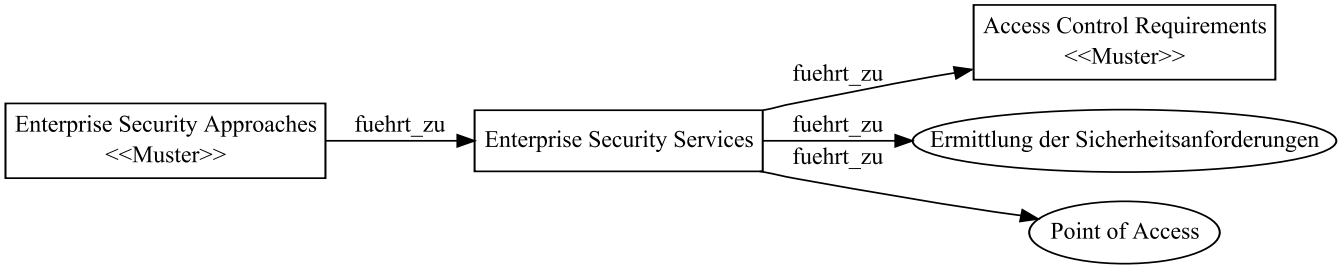
Gruppe(n)

- Analyse von Sicherheitsproblemen und Lösungsansätzen

Kurzbeschreibung

Das Pattern "Enterprise Security Services" baut auf dem Pattern "Enterprise Security Approaches" auf und beschreibt die Auswahl von Sicherheitsdiensten zur Umsetzung der ausgewählten Arten der Sicherung.

Beziehungen - Grafik




Beziehungen - Detail

- **Enterprise Security Approaches -> Enterprise Security Services** : Input: Lösungsansätze für Risiken
  Beziehung siehe S. 60, 61 in Security Patterns, Schumacher et al.
- **Enterprise Security Services -> Access Control Requirements** : als Autorisierungsdienst
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: *„Primäre Beispiele für solche Dienste sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“* (übersetzt aus dem Englischen)
  Quelle: Security Patterns, Schumacher et al.; S. 267: „*Basierend auf den Ergebnissen der Anwendung von Enterprise Security Services, ...“* (übersetzt aus dem Englischen)
  Die Spezifikation von Access Control Requirements führt zu einer Feinspezifikation der für die Autorisierung spezifizierten Inhalte aus der Anwendung des Enterprise-Security-Services-Patterns.
- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** : Informationsbasis
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: *„Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“* (übersetzt aus dem Englischen)
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Security Services -> Point of Access** : Gestaltung eines sicheren Zugriffspunkts für interne Applikationen
  Um einen sicheren Zugriffsweg auf interne Applikationen zu gestalten sollte ein Pattern der Gruppe Point of Access angewendet werden.

## *64. Entity Abstraction*

Quellen

**SOA Design Patterns, Erl**

S. 175-181

SOA Design Patterns, Thomas Erl, Prentice Hall, 3. Auflage, Februar 2010.

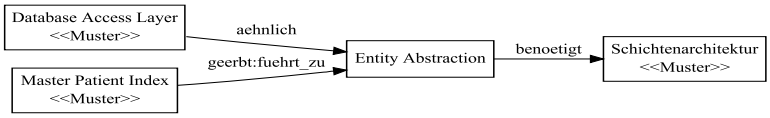
Schwerpunkt(e)

Gruppe(n)

Kurzbeschreibung

Das Pattern "Entity Abstraction" beschreibt beschreibt die Einführung einer zusätzlichen Schicht von Diensten, die den Zugriff auf sogenannte "Business Entities" (z. B. Kunden, Patienten, Produkte etc.), die häufig und in verschiedenen Prozessen verwendet werden, kapselt.

Beziehungen - Grafik




Beziehungen - Detail

- **Database Access Layer -> Entity Abstraction** : Hohe Ähnlichkeit
  Das Database-Access-Layer-Pattern weißt Ähnlichkeit mit dem Entity-Abstraction-Pattern aus SOA Design Patterns, Erl, S. 179 auf.

## 65. Entity-Attribute-Value

Quellen

**Paper: ArchiMed: a medical information and retrieval system**

Punkt 3.4 (S. 12 in Online-Quelle) beschreibt das Datenmodell, einen dort noch nicht so genannten generischen Entity-Attribute-Value Ansatz.

ArchiMed: a medical information and retrieval system; W. Dorda, T. Wrba, G. Duftschmid, P. Sachs, W. Gall, C. Rehnelt, G. Boldt, W. Premauer; Methods Inf Med; Vol. 38 (1), S. 16-24; 1999
Online verfügbar unter: http://www.meduniwien.ac.at/msi/mias/papers/Dorda1999a.pdf

**Paper: CEN prEN 13606 compliant export of medical data**

Diese Quelle belegt, dass das in der Quelle "Paper: ArchiMed: a medical information and retrieval system" beschriebene Datenmodell eine Spezialform von Entity-Attribute-Value ist. *"ArchiMed [4] is a clinical trial system developed at the Medical University of Vienna's Core Unit for Medical Statistics and Informatics. It allows to input and to statistically analyse data and further supports patient recruiting and the interactive design of case report forms (CRFs). ArchiMed uses the generic Entity-Attribute-Value (EAV) database design often used with medical data."*

CEN prEN 13606 compliant export of medical data from an Entity-Attribute-Value based patient record system; C. Rinner, T. Wrba, G. Duftschmid; TELEMED 2007; Berlin, Germany; 2007
Inhalt online verfügbar: http://www.meduniwien.ac.at/msi/mias/papers/Duftschmid2007a.pdf

**Paper: Exploring Performance Issues for a Clinical Database Organized Using an Entity-Attribute-Value Representation**

Exploring Performance Issues for a Clinical Database Organized Using an Entity-Attribute-Value Representation; Roland S Chen, Prakash Nadkarni, Luis Marenco, Forrest Levin, Joseph Erdos, Perry L Miller JAMIA 2000;7:475-487 doi:10.1136/jamia.2000.0070475

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Datenmodell-Abbildung

Kurzbeschreibung

Das Pattern "Entity-Attribute-Value" beschreibt einen Lösungsansatz, mit dem in relationalen Datenbanken dynamisch (um weitere Spalten) erweiterbare Entitäten gespeichert werden können.

### Kontext

Darstellung dynamischer Datenstrukturen (variable Feldmengen pro Datensatz) mittels relationaler Datenbanken.

### Problem

Tabellen in relationalen Datenbanken können die Menge der verfügbaren Attribute nicht für jede Zeile einzeln variieren.

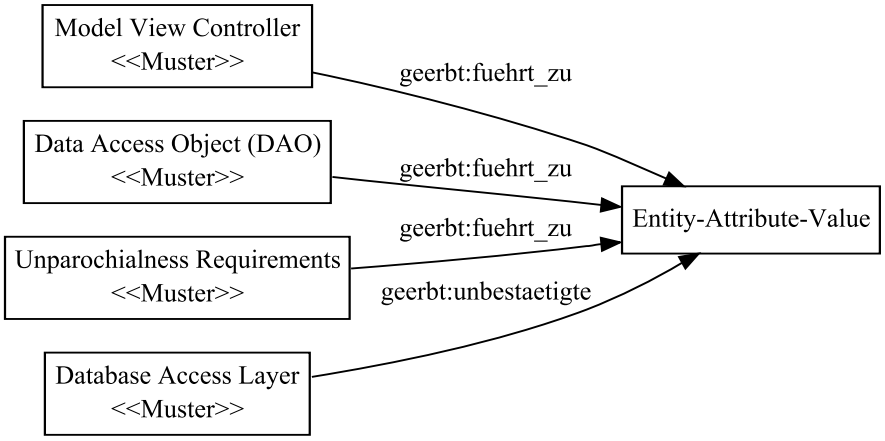
### Lösung

Als Lösung für das beschriebene Problem kann eine Tabelle mit den Spalten

| **Entity-Type** | **Key** | **Attribute** | **Value** |
|---|---|---|---|
| z. B. Patient | 1234567 | Name | Meierhöfer |

verwendet werden.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Model View Controller -> Datenmodell-Abbildung** : Gestaltung des Bestandteils Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Der Bestandteil "Model" kann in verschiedenen Formen abgebildet werden. Die gängigste Form dabei ist das Domain Model. Das Domain-Model-Pattern ist Bestandteil der Gruppe Datenmodell-Abbildung. Auch die anderen Elemente der Gruppe können zur Umsetzung eines Model im Rahmen der Anwendung des Model-View-Controller-Pattern verwendet werden.
  Quelle: S. 330; Patterns of Enterprise Application Architecture; Fowler;
  *“In its most pure OO form the model is an object within a Domain Model. You might also think of* [...] *“*
- **Data Access Object (DAO) -> Datenmodell-Abbildung** : Verwendet eine Ausprägung von

Geerbte Beziehung: fuehrt_zu_beziehung
Das DAO-Pattern dient der Vereinfachung des Zugriffs auf Datenbestände. Diese Datenbestände selbst können entsprechend der Prinzipien eines der Patterns der Gruppe Datenmodell-Abbildung aufgebaut sein.

- **Unparochialness Requirements -> Datenmodell-Abbildung** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Unparochialness-Requirements-Pattern dient der Definition von Anforderungen bezüglich der Fähigkeit eines Systems, sich veränderlichen Umgebungsbedingungen anzupassen. Die verschiedenen Patterns der Gruppe Datenmodell-Abbildung führen zu unterschiedlich flexiblen Datenmodellen. Während z. B. das Entity-Attribute-Value-Pattern zu einem leicht veränderbaren Datenmodell führt, wird beim Domain-Model-Pattern das Datenmodell relativ statisch, dafür aber einfacher verwendbar, abgelegt.
- **Database Access Layer -> Datenmodell-Abbildung** : Abhängig
  Geerbte Beziehung: unbestaetigte_beziehung
  Die konkrete Gestaltung der Database Access Layer ist abhängig von der gewählten Form der Datenmodell-Abbildung.

## *66. Equitable Resource Allocation*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 191-192

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

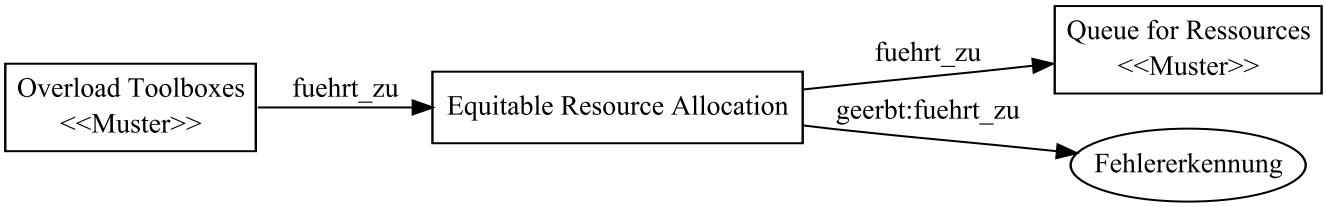
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Equitable Resource Allocation" beschreibt Empfehlungen für das angemessene Sperren von Ressourcen in Anwendungsfällen, in denen Ressourcen unterschiedlicher Queues auf gleiche Ressourcen zugreifen und sich dadurch gegenseitig aussperren können.

Beziehungen - Grafik




Beziehungen - Detail

- **Overload Toolboxes -> Equitable Resource Allocation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Equitable Resource Allocation -> Queue for Ressources** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *67. Error Containment Barrier*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 91-93

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

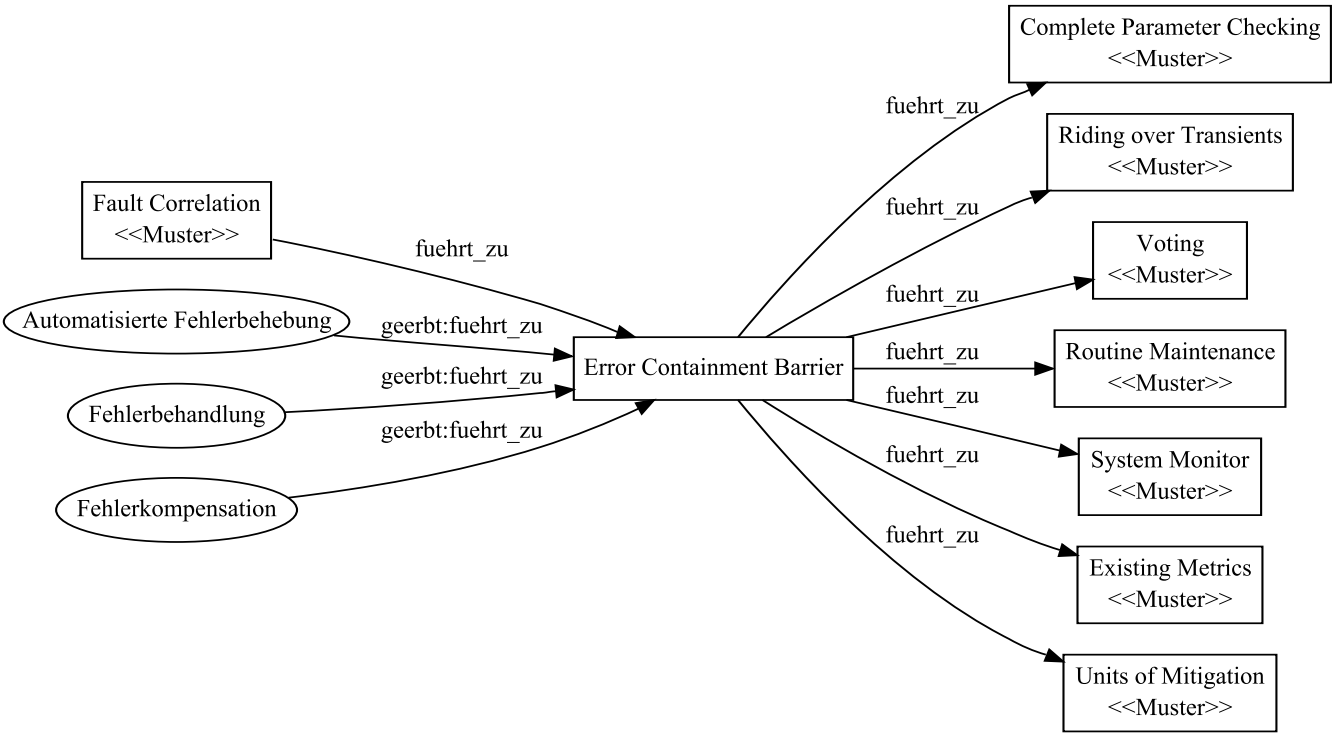
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Error Containment Barrier" beschreibt einen Weg, die Auswirkungen eines Fehlers auf genau eine Unit of Mitigation zu begrenzen.

Beziehungen - Grafik




Beziehungen - Detail

- **Fault Correlation -> Error Containment Barrier** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Complete Parameter Checking** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87, Abbildung 28;
- **Error Containment Barrier -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Routine Maintenance** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> System Monitor** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Units of Mitigation** : Wirkungsbereich
  Eine Error Containment Barrier dient dazu, die Auswirkung eines Fehlers auf eine Unit of Mitigation zu beschränken.
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 92
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu

auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *68. Error Correcting Codes*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 224-225

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

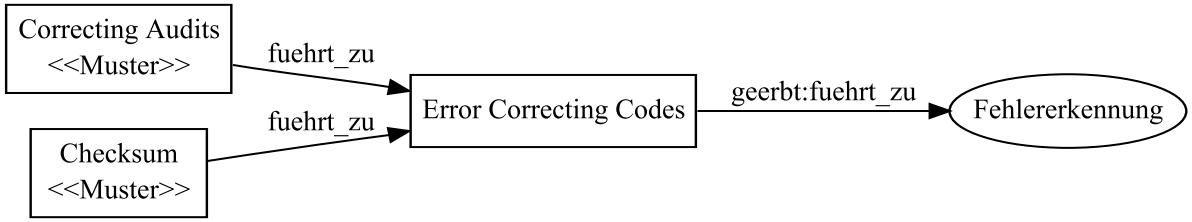
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Error Correcting Codes" subsumiert alle Verfahren, bei denen Prüfsummen ausreichend Information beinhalten, um eine automatische Korrektur ermittelter Fehler zu ermöglichen.

Beziehungen - Grafik



Beziehungen - Detail

- **Correcting Audits -> Error Correcting Codes** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Checksum -> Error Correcting Codes** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *69. Error Handler*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 148-150

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

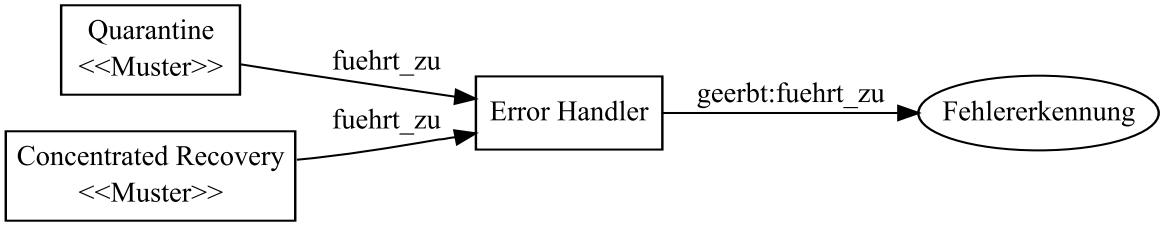
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Error Handler" beschreibt den Ansatz, Code für die Fehlerbehandlung in separaten Code-Blocks zusammenzufassen. Im einfachsten Fall ist das bereits durch try-catch-Blocks verbreiteter Programmiersprachen erreicht.

Beziehungen - Grafik



Beziehungen - Detail

- **Quarantine -> Error Handler** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Concentrated Recovery -> Error Handler** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *70. Escalation*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 69-72

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

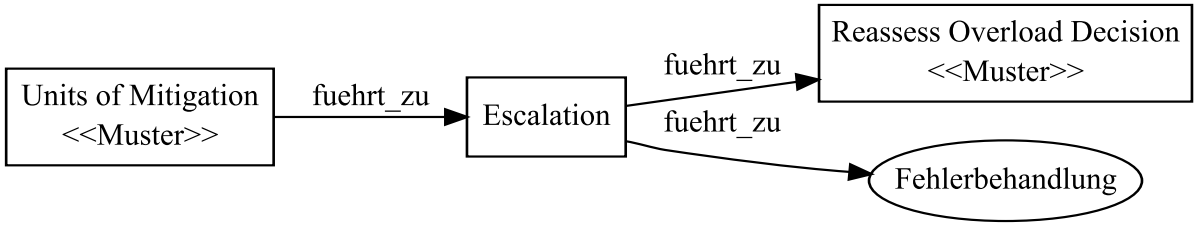
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Escalation-Pattern beschreibt das Vorgehen, die Fehlerbehandlung im Misserfolgsfall schrittweise von einfachen Maßnahmen mit geringen Nebenwirkungen hin zu besonders wirksamen Maßnahmen zu eskalieren.

Beziehungen - Grafik



Beziehungen - Detail

- **Units of Mitigation -> Escalation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Escalation -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Escalation -> Fehlerbehandlung** : Schrittweise Eskalation der Methoden
  Das Escalation-Pattern dient dazu, die Fehlerbehandlung so zu planen, dass sie beginnend bei einer möglichst einfachen, wenig aufwendigen Methode im Misserfolgsfall zu weiteren jeweils aufwendigeren Methoden eskaliert wird.
  Quelle: Patterns for Fault Tolerant Software; Robert S. Hanmer; S. 69;

## *71. Existing Metrics*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 116-117

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

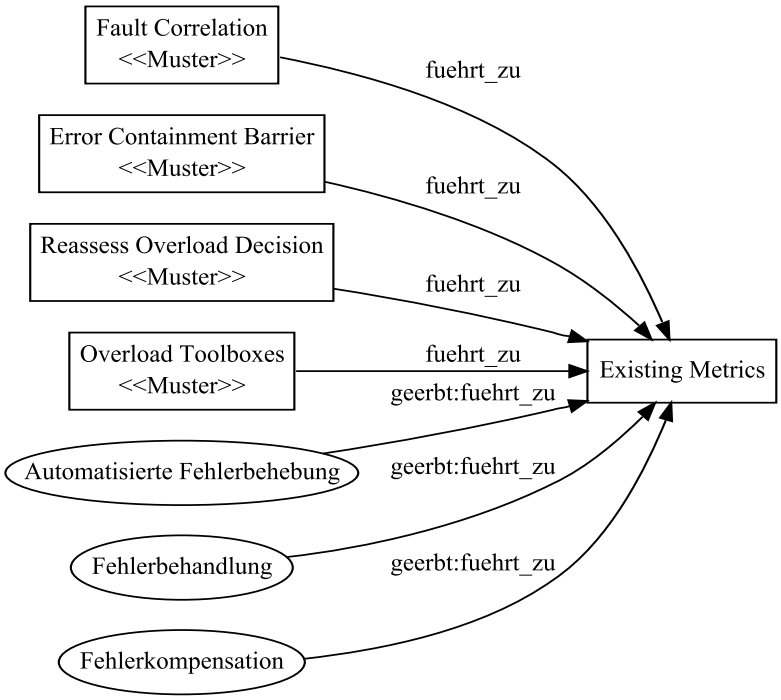
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Existing Metrics" beschreibt die Vorteilhaftigkeit der Verwendung von bereits aus anderen Gründen existierenden Werten als Indikatoren für die Ermittlung der Systemlast, da auf diese Weise keine zusätzliche Last zur Messung der Systemlast entsteht.

Beziehungen - Grafik




Beziehungen - Detail

- **Fault Correlation -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Reassess Overload Decision -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Overload Toolboxes -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte

Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## 72. Expansive Automatic Controls

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 195-197

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

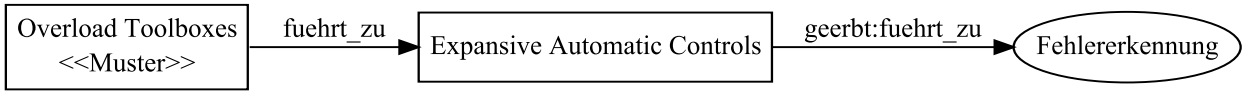
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Expansive Automatic Controls" beschreibt eine Strategie, die ein System befähigen soll, kurzfristige Überlastszenarios zu kompensieren, ohne Aufgaben in eine Warteschlange zu verschieben bzw. zu verwerfen.

Beziehungen - Grafik




Beziehungen - Detail

- **Overload Toolboxes -> Expansive Automatic Controls** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## 73. Extendability Requirements

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 246-253

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

### Schwerpunkt(e)

- Flexibilität

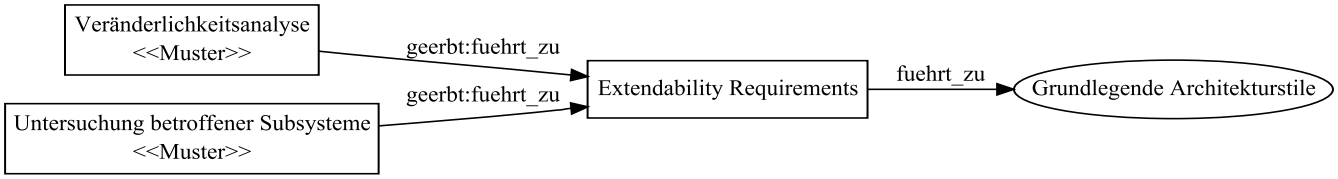
### Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

### Kurzbeschreibung

Das Pattern "Extendability Requirements" erläutert ein Verfahren zur Definition von Anforderungen an die Erweiterbarkeit eines Softwaresystems.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *74. Facade*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 185-193

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

**Patterns of Enterprise Application Architecture, Fowler**

Spezielle Ausprägung: Remote-Facade: S. 388-400

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

Facade: S. 294-295
Spezielle Ausprägung: Wrapper-Facade: S. 459-460

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Flexibilität

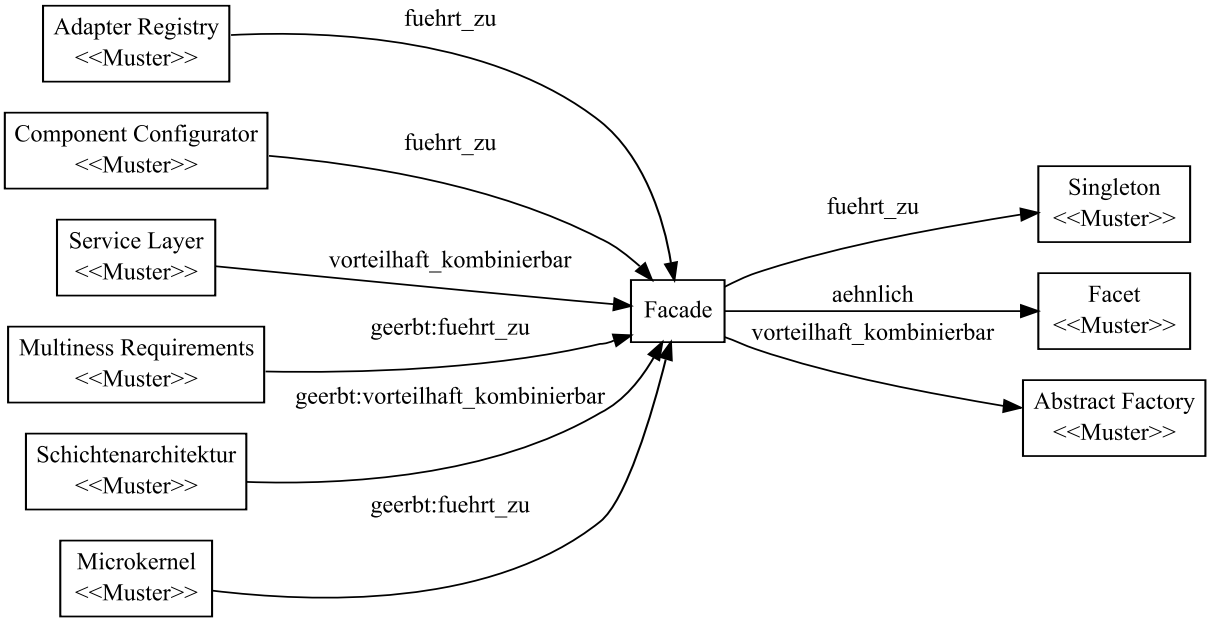
Gruppe(n)

- Kopplung und Entkopplung
- statische Veraenderung der Schnittstelle

Kurzbeschreibung

Das Pattern "Facade" oder "Fassade" beschreibt den Ansatz die Komplexität eines Programmteils (wie z. B. einer Komponente) hinter einer Menge gut verständlicher Methoden zu verbergen. Bezüglich seiner technischen Umsetzung ist dieses Pattern eng mit dem Adapter-Pattern verwandt.

Beziehungen - Grafik




Beziehungen - Detail

- **Adapter Registry -> Facade** : Adapter-Schnittstelle als Fassade der Registry-Klasse
  Die Adapter-Registry-Klasse verbirgt die Komplexität des Zugriffs auf eine Vielzahl von Adaptees mittels einer ebenso großen Zahl von Adaptoren durch die Bereitstellung aller Methoden der Adapter-Schnittstelle als Fassade der Adapter-Registry-Klasse.
- **Component Configurator -> Facade** : Verbergen der Komponentenschnittstelle
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Service Layer -> Facade** : Als Remote Facade
  Quelle: Patterns fo Enterprise Application Architecture; Fowler; S. 135
  *"Add remotability when you need it (if ever) by putting Remote Facades (388) on your Service Layer* [...]*"*
- **Facade -> Singleton** : Eine Instanz
  Um von einer Facade nur eine Instanz zu haben wird ein Singleton benötigt. (Siehe Gamma letzte Seite, Design Pattern Relationships).
  Quelle: Design Patterns, Gamma et al.; S. 193
  *Usually only one Facade object is required. Thus Facade objects are often Singletons.*
- **Facade -> Facet** : Verwechslungsgefahr
  Abgrenzung zu Fassade: Fassade bietet eine andere Schnittstelle, Facet eine bezüglich der Anzahl der Methoden verkleinerte Schnittstelle.
  Siehe http://c2.com/cgi/wiki?FacetPattern
- **Facade -> Abstract Factory** : Schnittstelle zur Generierung von Subsystem-Objekten
  Quelle: Design Patterns; Gamma et al.; S. 193
  *"Abstract Factory can be used with Facade to provide an interface for creating subsystem objects in a subsystem-independent way."*
- **Multiness Requirements -> statische Veraenderung der Schnittstelle** : Indirekt: Abbildung auf Ziel-Schnittstellen

Geerbte Beziehung: fuehrt_zu_beziehung
Grundsätzlich werden Muster zur statischen Veränderung der Schnittstelle verwendet um einzelne der Multiness Requirements zu erfüllen. Es ist aber zusätzlich ein größeres Pattern (Ebene Architektur) wie z. B. das Multi-Channel-Access-Provider-Pattern notwendig um die Multiness Requirements in einer konsistenten Architektur abzubilden.

- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *75. Facet*

Quellen

**Portland Pattern Repository**

http://c2.com/cgi/wiki?FacetPattern

Einstiegs-URL des Portland Pattern Repository: http://c2.com/ppr/

Schwerpunkt(e)

- Flexibilität

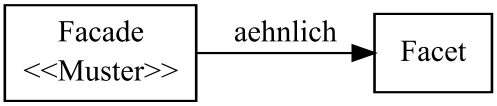
Gruppe(n)

- Handle Body Patterns

Kurzbeschreibung

Der Zweck des Patterns "Facet" ist es, eine einer Fassade ähnliche Lösung bereitzustellen, die eine verkleinerte Schnittstelle - also eine Auswahl bestimmter Methoden - zur Verfügung stellt.

Beziehungen - Grafik



Beziehungen - Detail

- **Facade -> Facet** : Verwechslungsgefahr
  Abgrenzung zu Fassade: Fassade bietet eine andere Schnittstelle, Facet eine bezüglich der Anzahl der Methoden verkleinerte Schnittstelle.
  Siehe http://c2.com/cgi/wiki?FacetPattern

## *76. Factory Method*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 107-116

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

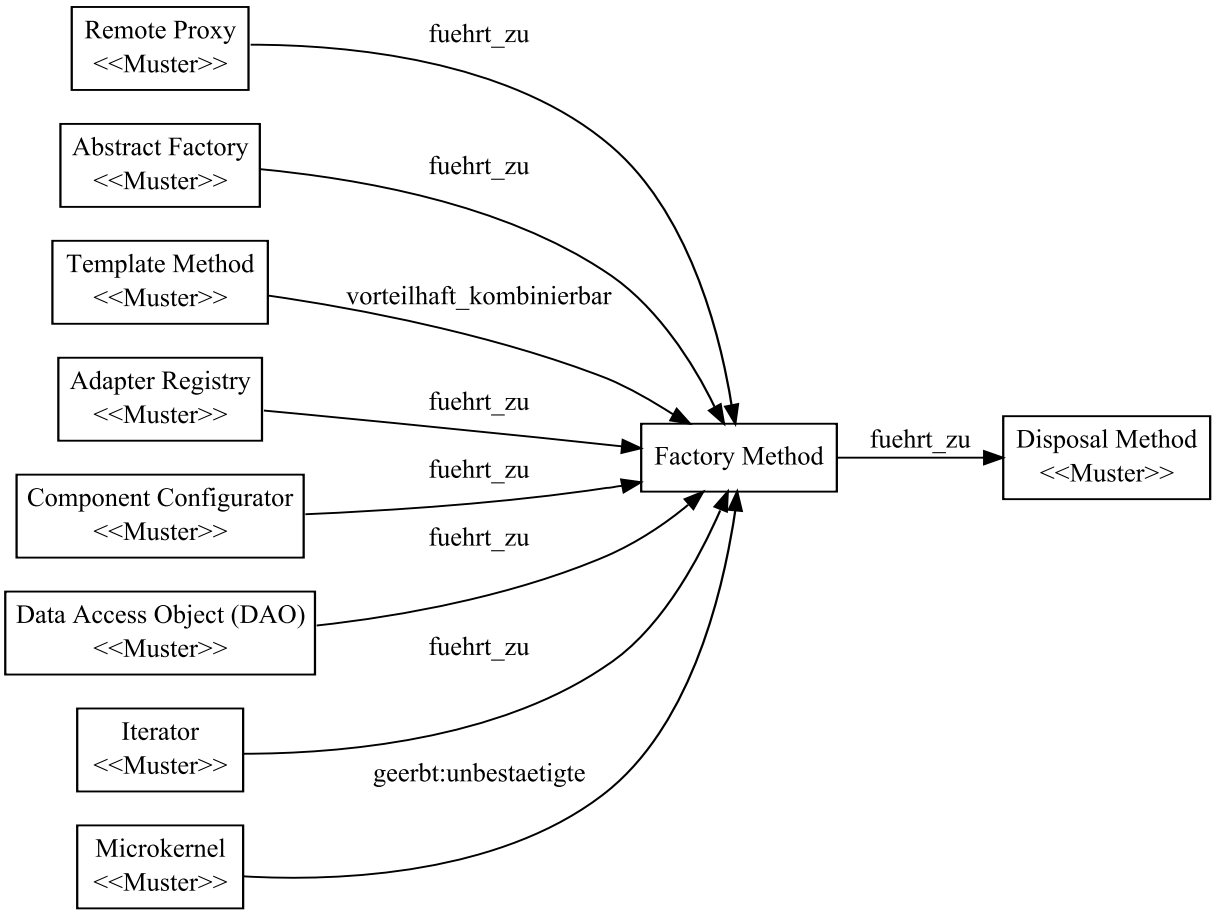
Gruppe(n)

- Instanziierung und Kontrolle der Instanzen

Kurzbeschreibung

Unter einer Factory Method versteht man eine Methode, die die Generierung von Objekten kapselt.

Beziehungen - Grafik




Beziehungen - Detail

- **Remote Proxy -> Factory Method** : Instanzierung
  Remote Proxy benötigt Lookup oder Factory zur Instanzierung wenn nicht bei jeder Instanzierung vollständige Verbindungsdaten zur entfernten Implementierung als Parameter übergeben werden sollen.
- **Abstract Factory -> Factory Method** : Umsetzbar unter Zuhilfenahme von Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Template Method -> Factory Method** : Benutzt häufig
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Adapter Registry -> Factory Method** : Instanzierung der Adaptoren durch Registry
  Die Instanzierung von Adaptoren muss z. B. im Falle des Aufrufs der reload()-Methode durch die Registry erfolgen. Da zur Instanzierung der Adaptoren immer auch die Auswahl des richtigen Konfigurationsobjekts bzw. die Instanzierung des

Konfigurationsobjekts mit anschließendem Laden der Konfigurationsdaten einher geht ist die Verwendung von Factory-Methoden zur Instanzierung empfehlenswert.

- **Component Configurator -> Factory Method** : Laden einer Komponente
  Quelle: S. 491; Pattern oriented Software Architecture, Volume 4
- **Data Access Object (DAO) -> Factory Method** : Zur Umsetzung einer DAO Factory
  Quelle:
  http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html;
  Abschnitt "Related Patterns";
- **Iterator -> Factory Method** : Instanzierung geeigneter Iterator-Subklassen
  Quelle: Design Patterns; Gamma et al.; S. 271
  *"Factory Method (107): Polymorphic iterators rely on factory methods to instantiate the appropriate Iterator subclass."*
- **Factory Method -> Disposal Method** : dispose created objects
  Quelle: Paper: Patterns in Java; S. 5 unten
- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen
  Geerbte Beziehung: unbestaetigte_beziehung
  Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.
  Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

## *77. Failover*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 163-165

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

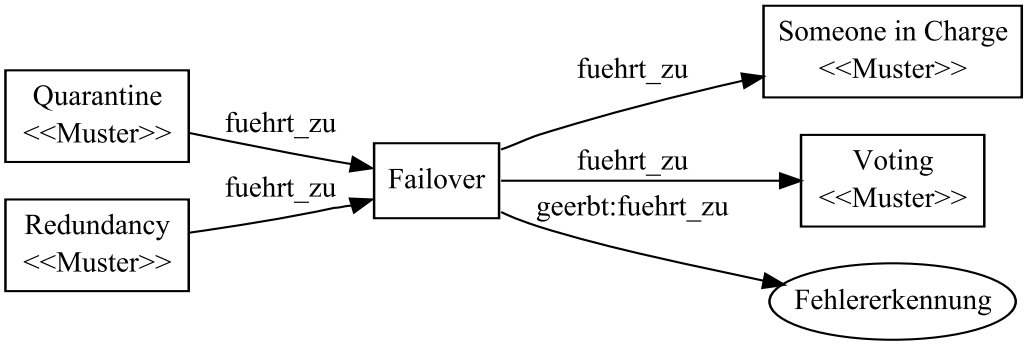
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Failover" beschreibt den Lösungsansatz, im Falle des Ausfalls eines Systembestandteils automatisch auf ein dazu redundantes Äquivalent auszuweichen.

Beziehungen - Grafik




Beziehungen - Detail

- **Quarantine -> Failover** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Redundancy -> Failover** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Failover -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Failover -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *78. Fault Correlation*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 88-90

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

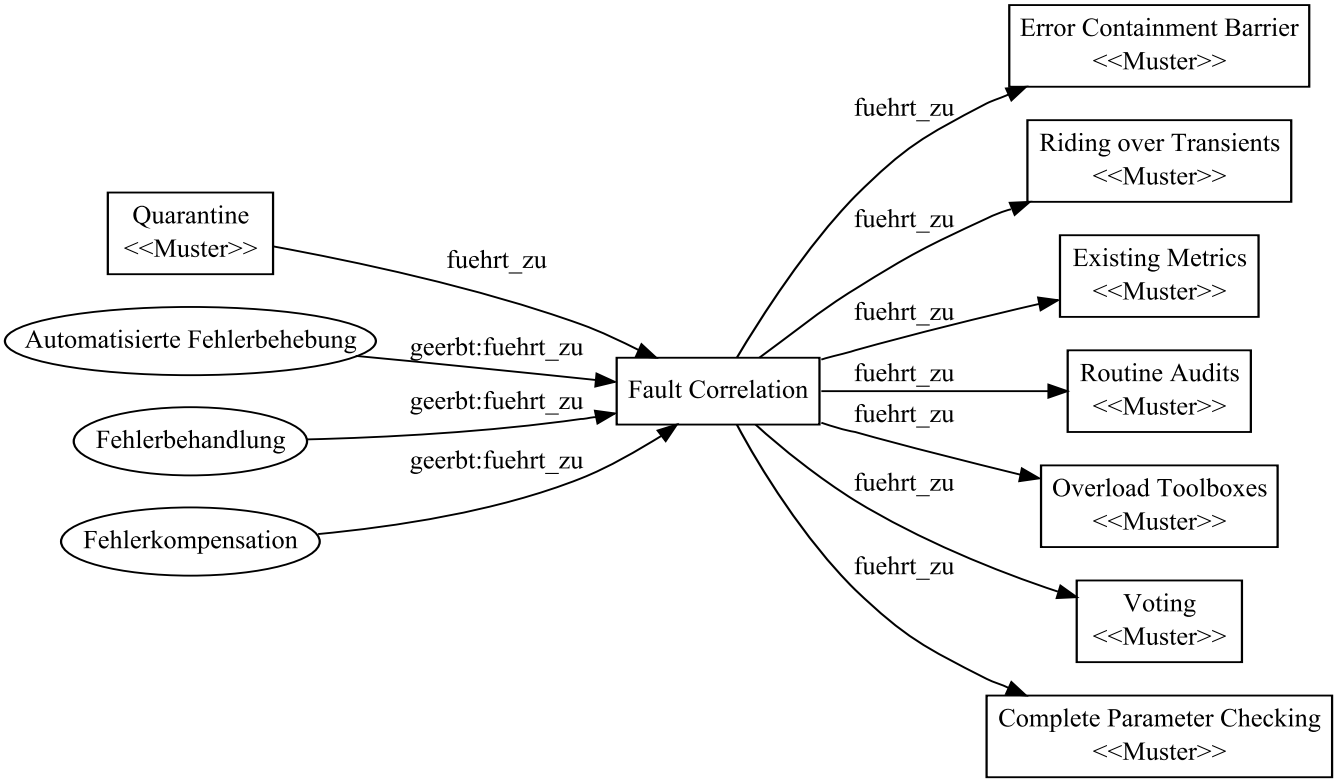
Gruppe(n)

- Fehlererkennung

## Kurzbeschreibung

Das Pattern "Fault Correlation" beschreibt ein Verfahren der Zuordnung von zum Design-Zeitpunkt nicht exakt bekannten Fehlern zu Fehlerkategorien, deren Behandlung bekannt und im System umgesetzt ist.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Quarantine -> Fault Correlation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fault Correlation -> Error Containment Barrier** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Fault Correlation -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Fault Correlation -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Fault Correlation -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Fault Correlation -> Overload Toolboxes** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fault Correlation -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Fault Correlation -> Complete Parameter Checking** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *79. Fault Observer*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S: 73-77

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

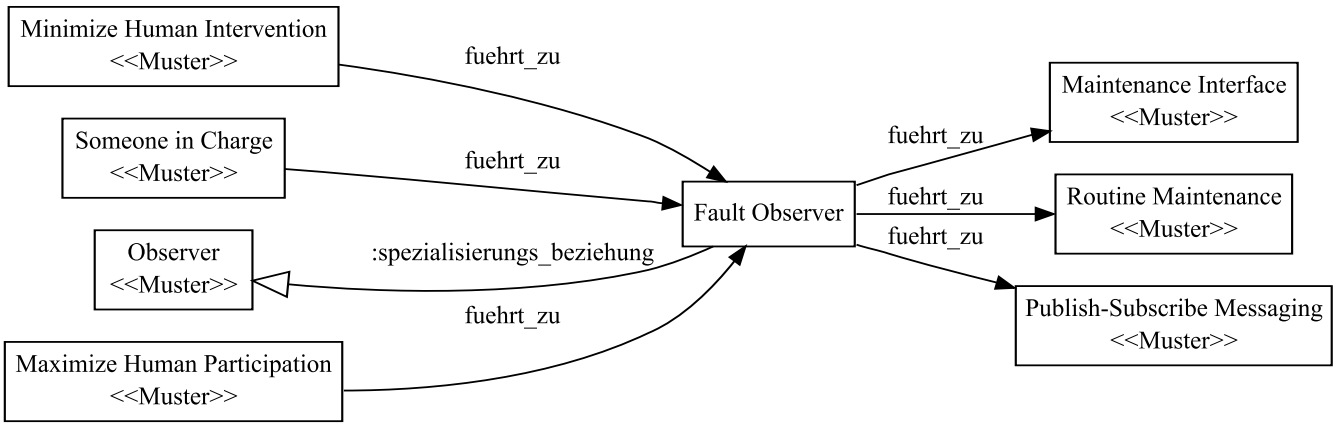
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Fault Observer" greift die Idee des Observer-Patterns auf und wendet diese etwas verallgemeinert auf die Fehlerbehandlung an. Kernbestandteil des Patterns ist ein Fehler-Überwachungsdienst, der eine oder mehrere andere Komponenten bezüglich des Auftretens von Fehlern überwacht.

Beziehungen - Grafik




Beziehungen - Detail

- **Minimize Human Intervention -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Someone in Charge -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Observer -> Fault Observer** : Spezialisierung: Fehlerüberwachung
  Anwendung der Prinzipien des Observer-Patterns auf die Überwachung von Fehlern.
- **Maximize Human Participation -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fault Observer -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Fault Observer -> Routine Maintenance** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fault Observer -> Publish-Subscribe Messaging** : Bestandteil der Umsetzung
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 74
  *"The Publischer-Subscriber pattern describes an effective way of distributing this fault information."*

## *80. File Authorization*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 350-354

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

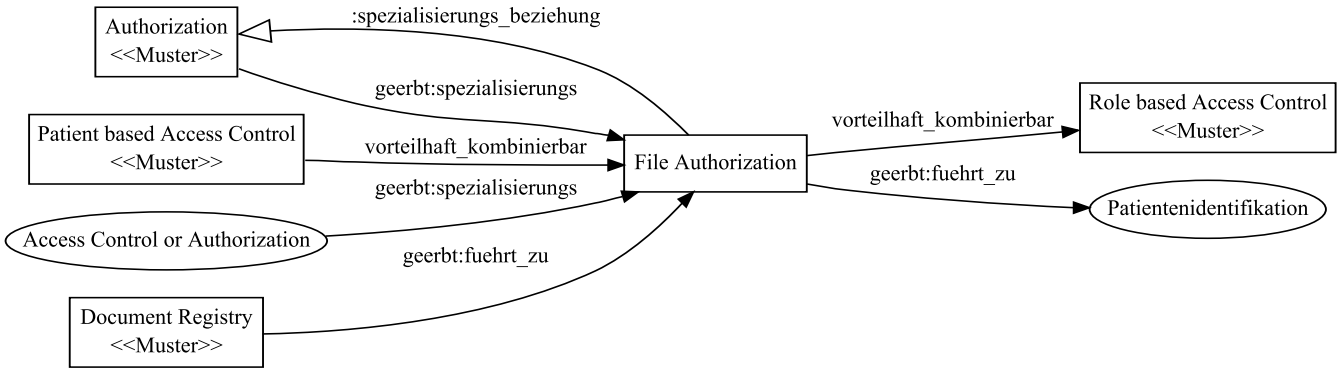
### Gruppe(n)

- Krankenaktenspezifische Berechtigungskonzepte

### Kurzbeschreibung

Das Pattern "File Authorization" beschreibt die Idee, Berechtigungen auf Ebene von Dateien bzw. Dokumenten zu vergeben.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Authorization -> File Authorization** : Granularität: Einzelne Datei
  Quelle: Security Patterns; Schumacher et al.; S. 351; Absätze Solution und Structure
- **Patient based Access Control -> File Authorization** : Mehrstufige Berechtigungen
  Patient based Access Control kann auch kombiniert mit File Authorization eingesetzt werden. So ist beispielsweise denkbar, den Zugriff auf Auflistungen vorhandener Dokumente durch Patient based Access Control zu realisieren, wärend das lesen oder ergänzen eines Dokuments durch File Authorization restriktiert wird.
- **File Authorization -> Role based Access Control** : Wenn Rollen genutzt werden
  Wenn für die Erteilung von Berechtigungen zum Dateizugriff Rollen verwendet werden sollen, ist das Role-based-Access-Control-Pattern notwendiger Bestandteil einer Lösung zur Umsetzung rollenbasierten Dateizugriffs.
  Quelle: Security Patterns; Schumacher et al.; S. 354
- **Access Control or Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Krankenaktenspezifische Berechtigungskonzepte sind spezielle Autorisierungsmechanismen. Sie leiten sich von allgemeinen Autorisierungsmechanismen ab.
- **Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Die Gruppe Krankenaktenspezifische Berechtigungskonzepte beinhaltet eine Menge von Patterns die spezifisch für den Einsatz in verteilten Krankenakten und eine Spezialisierung des Authorization-Patterns sind. (z. B. Case based Authorization)

- **Document Registry -> Krankenaktenspezifische Berechtigungskonzepte** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Kontext medizinischer Informationssysteme ist häufig bereits das Wissen über die Existenz eines Dokuments zu einem bestimmten Typ von Behandlung datenschutzrechtlich relevant. Aus diesem Grund ist es notwendig auch in einem Verzeichnisdienst, der die Suche über verteilt abgelegte Dokumente ermöglicht, ein umfassendes Berechtigungssystem zu integrieren.
- **Krankenaktenspezifische Berechtigungskonzepte -> Patientenidentifikation** :
  Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im medizinischen Kontext ist die zentrale Entität i.d.R. der Patient. Seine eindeutige Identifizierung, auch über Systemgrenzen hinweg, ist die Basis der meisten medizinspezifischen Autorisierungsmechanismen.

## *81. File Transfer*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 43

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

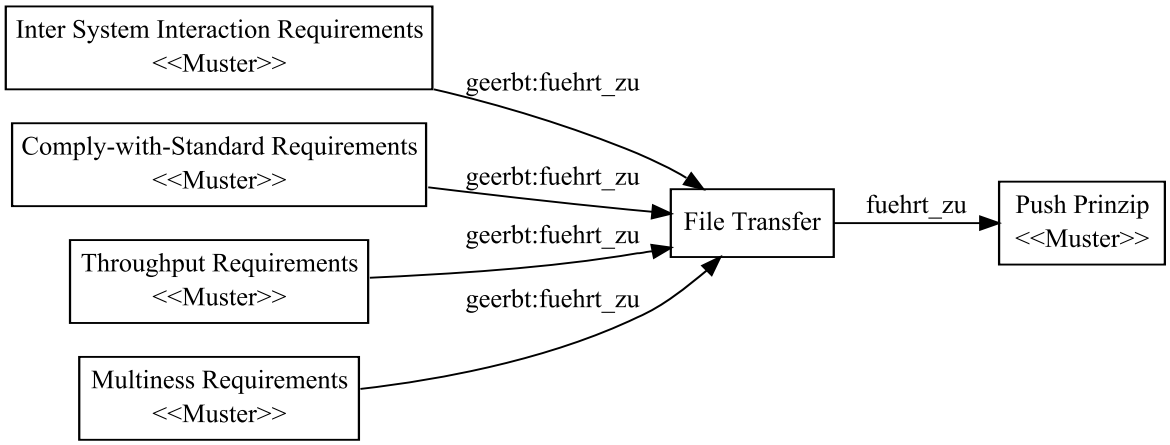
Gruppe(n)

- Grundlegende Integrationsformen

Kurzbeschreibung

Das Pattern "File Transfer" beschreibt die Datenübertragung mittels Datei-Export und Datei-Import.

Beziehungen - Grafik




Beziehungen - Detail

- **File Transfer -> Push Prinzip** :
  Datenübertragung per File-Transfer kann nur als aktives Ausliefern vom Sender an den oder die Empfänger (Push-Prinzip) realisiert werden.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** :
  Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** :
  beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.
- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards

auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## 82. Final Handling

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 203-205

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

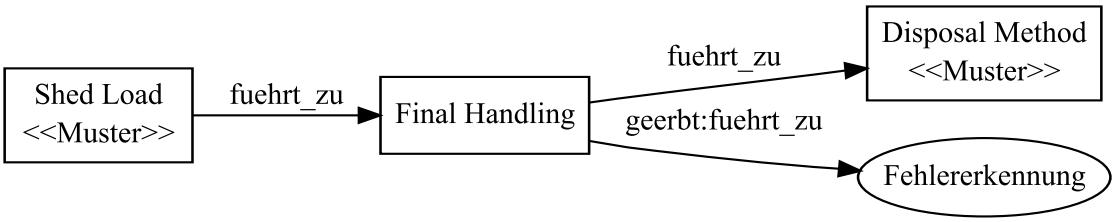
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Final Handling" beschreibt Ansatz, das Freigeben von reservierten Ressourcen sowohl im Fehlerfall als auch im Erfolgsfall durch die gleiche Routine zu behandeln. Es beschreibt somit das in vielen Programmiersprachen mittels der Schlüsselwörter try, catch und finally realisierte Vorgehen.

Beziehungen - Grafik




Beziehungen - Detail

- **Shed Load -> Final Handling** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Final Handling -> Disposal Method** : Kann durch Disposal Method realisiert werden
  Das Final-Handling-Pattern beschreibt das Freigeben belegter Ressourcen im Fehlerfall. Es kann sowohl in der Ausprägung "Separate Mechanism" als auch in der Ausprägung "Shared Mechanism" durch das Disposal-Method-Pattern realisiert werden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte

Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *83. Finish Work in Progress*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S: 214-216

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

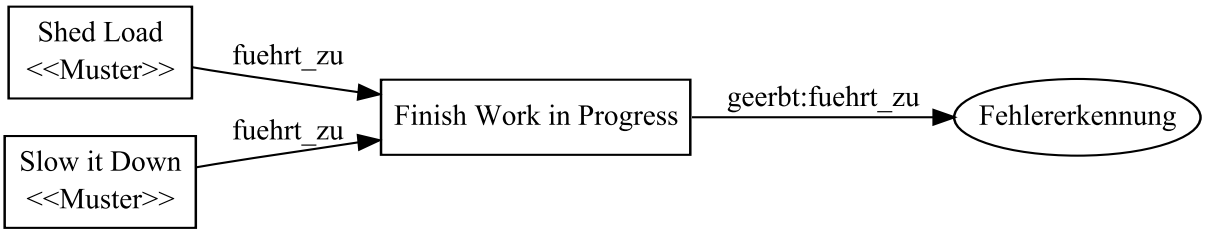
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Finish Work in Progress" bezieht sich auf die Verarbeitung komplexer Aufgaben, die aus mehreren einzelnen Schritten bestehen, die jeweils Ressourcen belegen und blockieren. Es sagt aus, dass Prozesse die bereits begonnen worden sind bei der Ressourcenbelegung Vorrang vor neuen Prozessen haben sollen, sodass begonnene Arbeit zuverlässiger und schneller abgeschlossen werden kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Shed Load -> Finish Work in Progress** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 202
  *"Finish work in progress discusses a way to intelligently select the work requests that should be rejected."*
- **Slow it Down -> Finish Work in Progress** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte

Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## 84. Flooding based Search

Schwerpunkt(e)

- Zuverlässigkeit
- Flexibilität

Gruppe(n)

- Suche im verteilten Dokumentensystem

Kontext

Suchen von Patienten oder Dokumenten über viele Systemknoten hinweg ohne zentralen Suchdienst.

Problem
Dokumente und/oder Patienten liegen auf verschiedenen Systemknoten vor. Ein zentraler Suchdienst ist nicht verfügbar oder kann die gewünschte Suche aufgrund von im Suchdienst fehlenden Metainformationen nicht eigenständig durchführen.

Lösung
Die Suche wird parallelisiert auf allen oder allen relevanten Systemknoten ausgeführt. Um in einer P2P-Umgebung alle Knoten zu erreichen, wird die Suchanfrage von Knoten zu Knoten weitergereicht. Genauer betrachtet reicht jeder Knoten, bei dem die Suchanfrage eintrifft, diese an alle ihm bekannten Knoten weiter und bearbeitet die Anfrage anschließend selbst. Knoten die die Anfrage bereits bearbeitet haben verwerfen diese und reichen sie entsprechend auch nicht weiter. Dieses Vorgehen ist notwendig, da der die Suche initialisierende Knoten unter Umständen nicht alle relevanten Knoten kennt.

Konsequenzen
- Nachteile

- hoher Zeitaufwand
- im P2P Fall ist nicht sichergestellt, dass tatsächlich alle Knoten durchsucht werden
- großer Einfluss/Risiko durch Ausfall von einzelenen Systemknoten

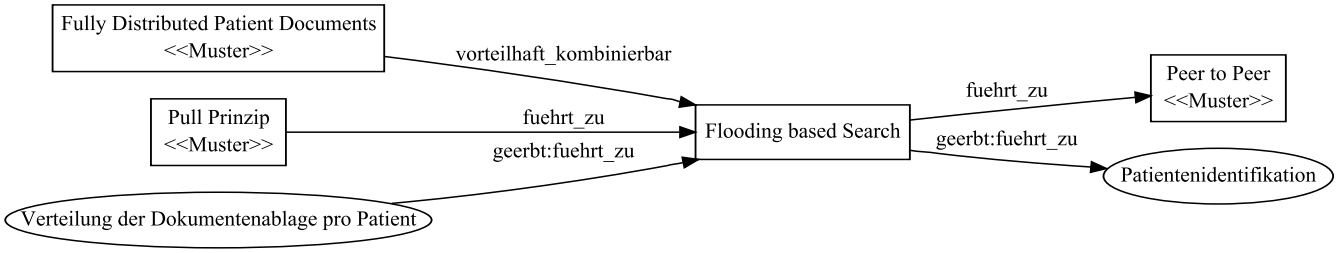
- Vorteile:

- Alle Dokumentenbestandteile stehen als Metadaten zur Suche zur Verfügung, d.h. auch die komplexest möglichen Suchen können prinzipiell ausgeführt werden

Quelle

Vassilios V. Dimakopoulos und Evaggelia Pitoura, “On the Performance of Flooding-Based Resource Discovery,” IEEE Trans. Parallel Distrib. Syst. 17, no. 11 (2006): 1242-1252.

Beziehungen - Grafik




Beziehungen - Detail

- **Fully Distributed Patient Documents -> Flooding based Search** : Findung von Dokumenten
  Alternativ zur Installation eines zentralen Suchdiensts kann in einem System mit verteilten Datenquellen auch ein verteilter Suchmechanismus realisiert werden. Basis für diese Lösung ist die aus dem Bereich der Peer-to-Peer-Netze bekannte Flooding-basierte Suche. Bei diesem Ansatz werden für jede Suchanfrage alle im System vorhandenen Knoten für jeweils zum Suchen innerhalb ihrer eigenen Daten verwendet.
  Dieser Mechanismus führt allerdings zu einer erhöhten, durch Suchen verursachten Last auf den einzelnen Systemknoten.
- **Pull Prinzip -> Flooding based Search** : Zur Findung abholbarer Dokumente
  Alternativ zur Verwendung einer zentralen Document Registry kann der Ablageort eines Dokuments auch über einen Flooding-based-Search-Mechanismus ermittelt werden. Dabei werden alle bekannten Ablageorte rekursiv nach Dokumenten oder spezifischen Suchkriterien durchsucht. Die dadurch erlangte Kenntnis über den Ablageort ist notwendig, um das Dokument aktiv abholen zu können.
- **Flooding based Search -> Peer to Peer** : Benötigt zwangsläufig
  Flooding ist eine ausschließlich im Peer-to-Peer-Bereich anwendbare Art der Suche.
- **Verteilung der Dokumentenablage pro Patient -> Suche im verteilten Dokumentensystem** : Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Wenn Dokumente verteilt abgelegt werden, ist es notwendig sie wiederfinden zu können.
- **Suche im verteilten Dokumentensystem -> Patientenidentifikation** : Zur eindeutigen Zuordnung der Suchergebnisse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um in einem verteilten Dokumentensystem gespeicherte Dokumente zu einem Patienten zu suchen, ist es notwendig, dass der Patient über die Grenzen der einzelnen Subsystem-Knoten hinweg eindeutig identifizierbar bleibt. Besonders Systeme mit heterogenen Patienten-IDs benötigen einen Mechanismus zur subsystemübergreifenden Patientenidentifikation.

## *85. Flyweight*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 195-206

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

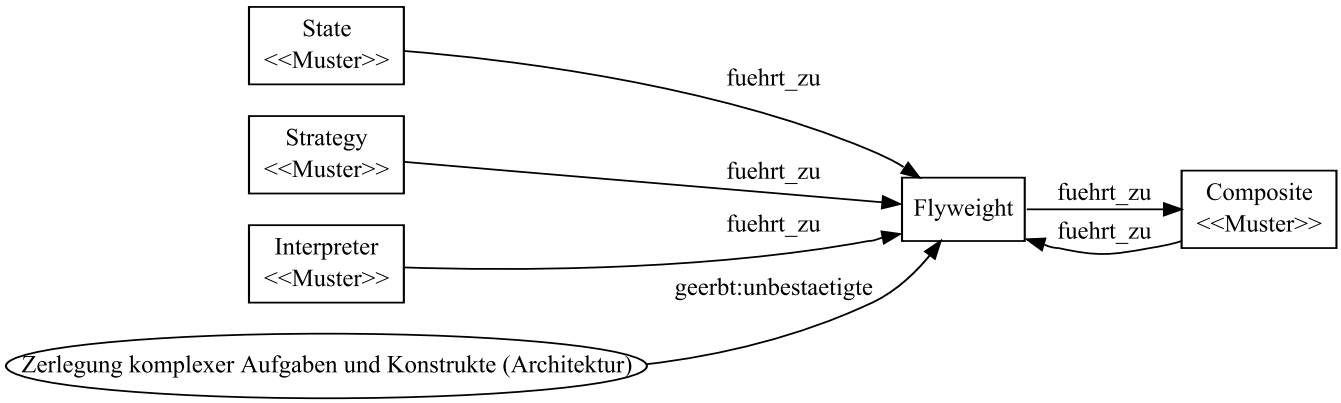
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

Kurzbeschreibung

Das Pattern "Flyweight" beschreibt einen Ansatz, die Anzahl notwendiger Objektinstanzen durch gemeinsame Nutzung zu reduzieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Composite -> Flyweight** : sharing composites
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **State -> Flyweight** : sharing states
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma et al.; S. 206;
  *“It's often best to implement State and Strategy objects as flywights”*
- **Strategy -> Flyweight** : sharing strategies
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma et al.; S. 206;
  *“It's often best to implement State and Strategy objects as flywights”*

- **Interpreter -> Flyweight** : sharing terminal symbols
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Flyweight -> Composite** :
  Quelle: Design Patterns; Gamma et al.; S. 206
  *"The Flywight pattern is often combined with the Composite pattern to implement a logically hierarchical structure in terms of a directed-acyclic graph with shared leaf nodes."*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *86. Fresh Work Before Stale*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S: 217-219

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

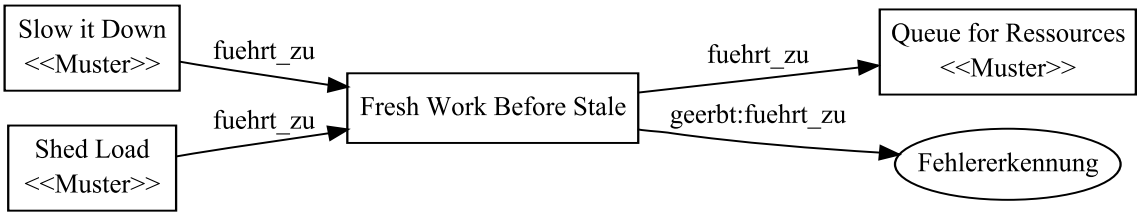
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Fresh Work Before Stale" beschreibt die Rahmenbedingungen, unter denen es besser ist, neu eintreffende Anfragen vor den bereits wartenden Anfragen zu bearbeiten.

Beziehungen - Grafik



Beziehungen - Detail

- **Slow it Down -> Fresh Work Before Stale** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Shed Load -> Fresh Work Before Stale** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Fresh Work Before Stale -> Queue for Ressources** : Als „Last In First Out“-Queue
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; S. 217-219
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *87. Front Door*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 473-479

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

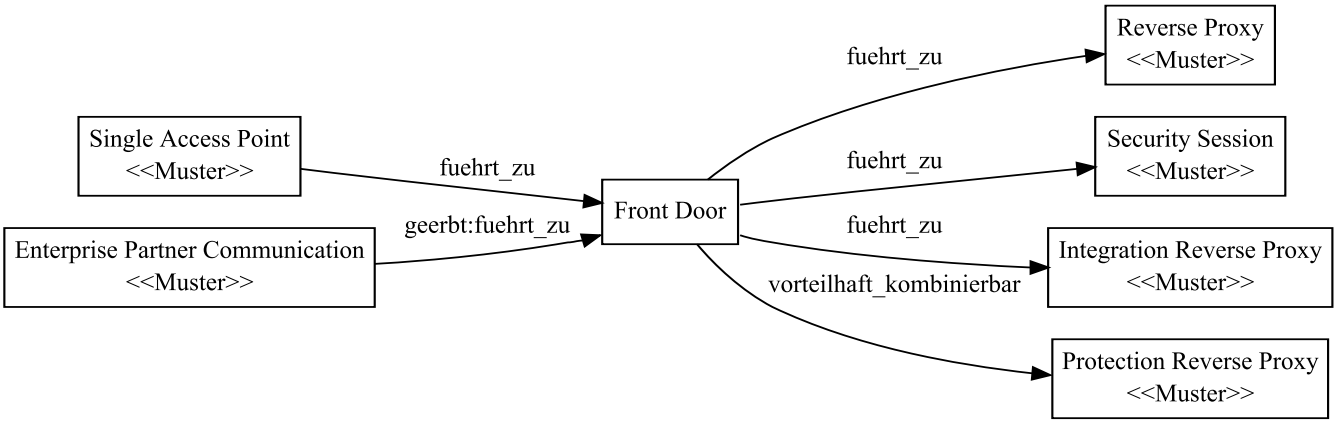
Gruppe(n)

- Access Restriction

Kurzbeschreibung

Das Pattern "Front Door" beschreibt eine Lösung zur zentralen Authentifizierung von Benutzern (z. B. an Webanwendungen) und baut dabei auf einer Menge anderer Patterns, wie z. B. dem Integration-Reverse-Proxy-Pattern auf.

Beziehungen - Grafik




Beziehungen - Detail

- **Single Access Point -> Front Door** : Agiert als
  Der Integration Reverse Proxy des Front-Door-Patterns agiert als einziger Zugriffsweg (Single Access Point) zu den dahinter liegenden Systemen.
  Quelle: Security Patterns; Schumacher et al; S. 479
  *"You can view Front Door as adding Checkpoint and Security Session to an Integration Reverse Proxy or Protection Reverse Proxy, and thus also providing a Single Access Point to a company's Web applications and services."*
- **Front Door -> Reverse Proxy** : Bestandteil
  Quelle: Security Patterns; Schumacher et al.; S. 473
  *"A reverse proxy is an ideal point to implement authentication and authorizadion, by implementing a Web entry server for your back-ends"*
  Quelle: Security Patterns; Schumacher et al.; S. 475
  *"Solution: Implement a Front Door server as a specialization of the Integration Reverse Proxy that identifies user and keeps track of user sessions.* [...] *,remember that it can also act as a Protection Reverse Proxy for the public part of the Web site."*
- **Front Door -> Security Session** : Verwendet
  Quelle: Security Patterns; Schumacher et al.; S. 304;
  *"Integration Reverse Proxy and Front Door rely on Security Session to keep track of Web users."*
- **Front Door -> Integration Reverse Proxy** : Bestandteil
  Der Integration Reverse Proxy ist ein Bestandteil des Front-Door-Patterns. Front Door kann aber auch als Erweiterung des Integration-Reverse-Proxy-Patterns versanden werden.
  Quelle: S. 472; Security Patterns; Schumacher et al.
- **Front Door -> Protection Reverse Proxy** :
  Quelle: Security Patterns; Schumacher et al.; S. 462;
  *"Integration Reverse Proxy (465) and Front Door (473) can (and should) be combined in their function with Protection Reverse Proxy (457), and thus vary this pattern by adding functionality."*
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR

Geerbte Beziehung: fuehrt_zu_beziehung
Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *88. Full Access with Errors*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 305-311

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

Unter dem Namen "Full View with Errors" auf S. 17

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

Schwerpunkt(e)

- Sicherheit

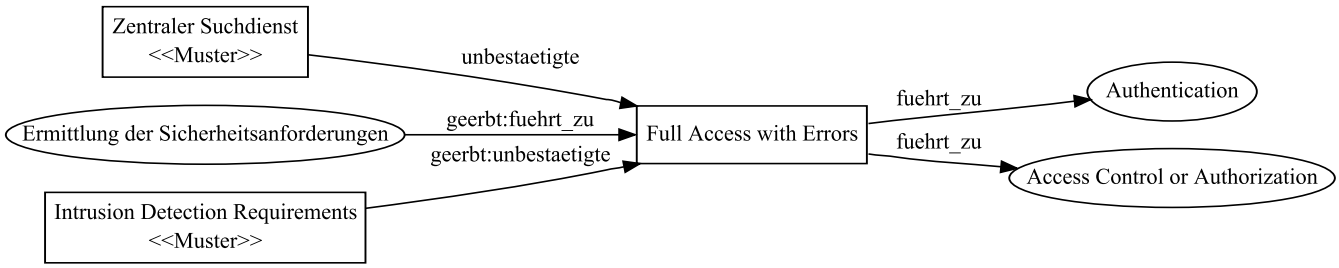
Gruppe(n)

- Type of Access

Kurzbeschreibung

Das Pattern "Full Access with Errors" beschreibt den Ansatz, in einer Anwendung alle verfügbaren Funktionen anzuzeigen und im Falle nicht vorliegender Berechtigungen eine Fehlermeldung auszugeben.

Beziehungen - Grafik




Beziehungen - Detail

- **Zentraler Suchdienst -> Full Access with Errors** : Steuerung der Zugriffsberechtigung auf Suchergebnisse
  Die Verwendung eines zentralen Suchdienstes führt dazu, dass entweder eine Full-

Access-with-Errors-Lösung in Verbindung mit dezentralen Autorisierungslösungen oder eine Limited-Access-Lösung mit zentralen Autorisierungslösungen denkbar ist.

- **Full Access with Errors -> Authentication** : Ermittlung des Benutzers
  Um einen durch Full Access with Errors beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
  Quelle: Security Patterns, Schumacher et al.; S: 309
  Beschreibt die Beziehung zu Identifcation und Authentication Patterns.
- **Full Access with Errors -> Access Control or Authorization** :
  Berechtigungsermittlung
  Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.
- **Intrusion Detection Requirements -> Type of Access** : beeinflusst vermutlich
  Geerbte Beziehung: unbestaetigte_beziehung

## *89. Fully Distributed Patient Documents*

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Verteilung der Dokumentenablage pro Patient

Kontext

Die Anforderungen an Zuverlässigkeit, Skalierbarkeit und andere Kriterien sind definiert. Die technische Umgebung des Systems wurde untersucht und dokumentiert. Die Ergebnisse dieser Untersuchung zeigen umfassende Kompetenz zum eigenständigen Betrieb von IT-Systemen bei allen beteiligten Einrichtungen. Eventuell besteht sogar ein nicht technisch bedingter Zwang zu Vermeidung einer zentralen Datenspeicherung.

Problem

Wie können Dokumente zu Patienten in einem verteilten System gespeichert werden, ohne einen zentralen Datenspeicher zu betreiben?

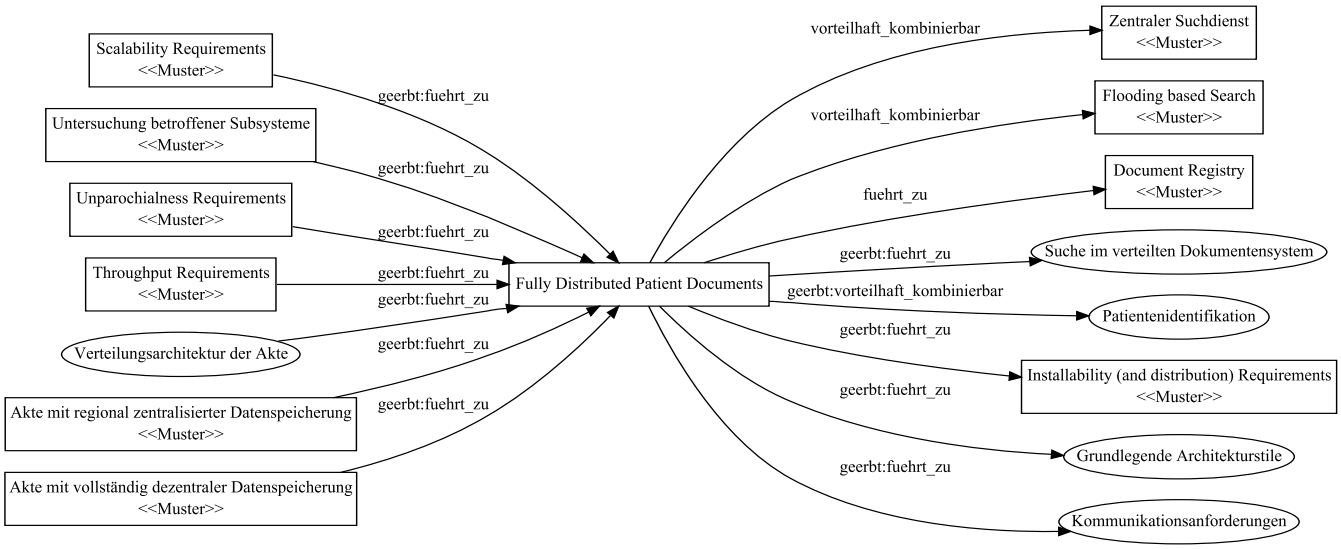
Grundsätzlich bestehen die Möglichkeiten:

- die Daten an einem pro Patient zentralen Punkt
- oder am Ort der Dokumentenentstehung (bzw. nahe diesem Ort) zu speichern.

Lösung

Vollständig versteilte Speicherung liegt dann vor, wenn die Speicherung der Dokumente am Ort ihrer Entstehung erfolgt. Dadurch kann die Speicherung ohne zentrale Datenspeicher erreicht werden.

Beziehungen - Grafik




Beziehungen - Detail

- **Fully Distributed Patient Documents -> Zentraler Suchdienst** : Findung von Dokumenten
  Um Dokumente in einem System aus einer Vielzahl von Datenquellen wieder finden zu können, kann ein zusätzlicher Knoten in das System eingefügt werden, der als zentraler Suchdienst agiert. Ein zentraler Suchdienst muss in diesem Fall entweder von den verteilten Knoten, die Dokumente speichern über die Entstehung neuer Dokumente informiert werden, oder diese in regelmäßigen Abständen nach neuen Dokumenten durchsuchen. Auf diese Weise kann ein Index über die im Gesamtsystem verfügbaren Dokumente erstellt werden, der für die Umsetzung eines zentralen Suchdiensts notwendig ist.

  Umgekehrt ist für die Kombination aus Fully Distributed Patient Documents und Zentraler Suchdienst folgendes festzuhalten:
  Wird für die Verteilung der Dokumentenablage das Pattern Fully Distributed Patient Documents gewählt, so eignet sich als Registry für den Suchindex das Document-

Registry-Pattern. Alternativ kann auch eine Registry, die nur die Knoten, in denen Dokumente zu einem Patienten vorliegen verwaltet, verwendet werden. Nur durch einen Index, der alle Senken von Dokumenten zu Patienten zuordnet, kann vermieden werden, dass jeder Suchvorgang über alle Systemknoten hinweg ausgeführt werden muss.

- **Fully Distributed Patient Documents -> Flooding based Search** : Findung von Dokumenten
  Alternativ zur Installation eines zentralen Suchdiensts kann in einem System mit verteilten Datenquellen auch ein verteilter Suchmechanismus realisiert werden. Basis für diese Lösung ist die aus dem Bereich der Peer-to-Peer-Netze bekannte Flooding-basierte Suche. Bei diesem Ansatz werden für jede Suchanfrage alle im System vorhandenen Knoten für jeweils zum Suchen innerhalb ihrer eigenen Daten verwendet.
  Dieser Mechanismus führt allerdings zu einer erhöhten, durch Suchen verursachten Last auf den einzelnen Systemknoten.
- **Fully Distributed Patient Documents -> Document Registry** : zur Registrierung und Wiederfindung
  Wenn Dokumente verteilt im System auf verschiedenen Systemkonten (auch heterogenen Systemknoten) angelegt werden können ist eine zentrale Registrierstelle, an der der Ablageort jedes Dokuments vermerkt ist hilfreich.
- **Scalability Requirements -> Verteilung der Dokumentenablage pro Patient** : Skalierbarkeitsanforderungen beeinflussen Verteilung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Patterns der Gruppe Verteilung der Dokumentenablage pro Patient beschreiben Architekturvarianten für die Gestaltung der Ablage von Dokumenten. Die zentrale Aufgabe einer verteilten Krankenakte beinhaltet den regelmäßigen schreibenden und lesenden Zugriff auf Dokumente zu Patienten. Die Skalierbarkeitsanforderungen beschreiben, in welcher Form und welchem Umfang das System wachsen und schrumpfen können soll. Diese Fähigkeit basiert unter anderem auf der Architektur der Dokumentenablage. Deshalb wird diese durch die Skalierbarkeitsanforderungen direkt beeinflusst.
- **Untersuchung betroffener Subsysteme -> Verteilung der Dokumentenablage pro Patient** : Verwendbarkeit der Subsysteme
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Unparochialness Requirements -> Verteilung der Dokumentenablage pro Patient** : schränken ein
  Geerbte Beziehung: fuehrt_zu_beziehung
  Gewählte Unparochialness Requirements schränken die Menge der möglichen Verteilungslösungen ein. Abgeleitet aus (Withall, S. 260 Punkt 5, External Systems) -> Anforderungen an Kommunikationsfähigkeiten zu externen Systemen.
- **Throughput Requirements -> Verteilung der Dokumentenablage pro Patient** : Beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Beispiel:

Werden in einer verteilten Krankenakte Dokumente hauptsächlich in der Einrichtung ihrer Erstellung benutzt, so wird die maximale Kapazität für den Durchsatz durch vollständig dezentral gespeicherte Dokumente (Fully Distributed Patient Documents) erreicht. Dabei wird schon bei der Erstellung des Dokuments durch ortsnahe Speicherung und die daraus resultierende Nutzung lokaler Hochgeschwindigkeitsnetze bei der gleichzeitig keine Belastung für dritte Systeme entsteht, hoher Datendurchsatz ermöglicht. Da dritte Systeme nicht belastet werden, steht deren volle Leistungsfähigkeit ebenfalls für den lokalen Nutzungsfall zur Verfügung.
Verändert sich die Art der Dokumentennutzung allerdings hin zu einer verstärkten Nutzung von Dokumenten anderer Einrichtungen, so sinkt der mögliche Durchsatz deutlich, da Suchvorgänge über eine Vielzahl dezentraler Knoten notwendig werden. Diese Vorgänge belasten gleichzeitig viele Systemknoten, die dann für den lokalen Anwendungsfall nicht mehr ihre volle Leistungsfähigkeit zur Verfügung stellen können. In diesem Szenario entwickelt das andere Pattern der Gruppe Verteilung der Dokumentenablage pro Patient seine Vorteile, da hier auch für starke verteilte Nutzung pro Patient immer nur ein Systemknoten für Suchvorgänge belastet wird.

- **Verteilungsarchitektur der Akte -> Verteilung der Dokumentenablage pro Patient** : unterschiedlich kombinierbar
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Akte mit regional zentralisierter Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung einer Akte mit regional zentralisierter Datenspeicherung muss festgelegt werden, ob die Daten zu einem Patienten in einem, dem Patienten fest zugeordneten regionalen Knoten gespeichert werden oder in pro Patient möglicherweise mehreren Knoten abgelegt werden sollen. Der letztere Fall kann z. B. auftreten, wenn der Ablageort nicht dem Patienten regional zugeordnet, sondern der Einrichtung, die den Patienten behandelt regional zugeordnet wird.
- **Akte mit vollständig dezentraler Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Suche im verteilten Dokumentensystem** : Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Wenn Dokumente verteilt abgelegt werden, ist es notwendig sie wiederfinden zu können.
- **Verteilung der Dokumentenablage pro Patient -> Patientenidentifikation** :
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wenn Dokumente verteilt abgelegt werden ist es hilfreich, wenn die zugehörigen Patienten über die verschiedenen Systemknoten hinweg eindeutig identifizierbar sind.
- **Verteilung der Dokumentenablage pro Patient -> Installability (and distribution) Requirements** : Anforderungen zur verteilten Installation
  Geerbte Beziehung: fuehrt_zu_beziehung
  Je stärker die Dokumentenablage verteilt werden soll, desto wichtiger wird es, dass die

Softwaresysteme der Knoten auf denen die Dokumentenablage erfolgt, einfach zu installieren ist.

- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.

## *90. Half Object Plus Protocol*

Quellen

**Gerard Meszaros**

Half-Object + Protocol; Gerard Meszaros;in Pattern Languages of Program Design, vol 1., J.O. Coplien and D.C. Schmidt, eds. Addison-Wesley, 1995, S. 129-132

Schwerpunkt(e)

- Flexibilität

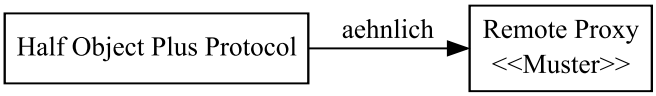
Gruppe(n)

- Handle Body Patterns

Kurzbeschreibung

Das Pattern "Half Object Plus Protocol" beschreibt die Zerlegung eines Objekts, das in zwei Adressbereichen eingesetzt werden muss, in zwei Hälften, die über ein definiertes Protokoll miteinander kommunizieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Half Object Plus Protocol -> Remote Proxy** : Lösen das gleiche Problem, komfortable entfernte Kommunikation
  Quelle: Portland Pattern Repository; http://c2.com/cgi/wiki?HalfObjectPlusProtocol
  Letztmalig geprüft: 13.10.2011
  Letzte Änderung laut Seite: 09.07.2003

In Beschreibung zu Half Object Plus Protocol: „*Patterns which solve the same problem include RemoteProxy (which delegates all requests back to a single site)*“

## *91. Heartbeat*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 101-103

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

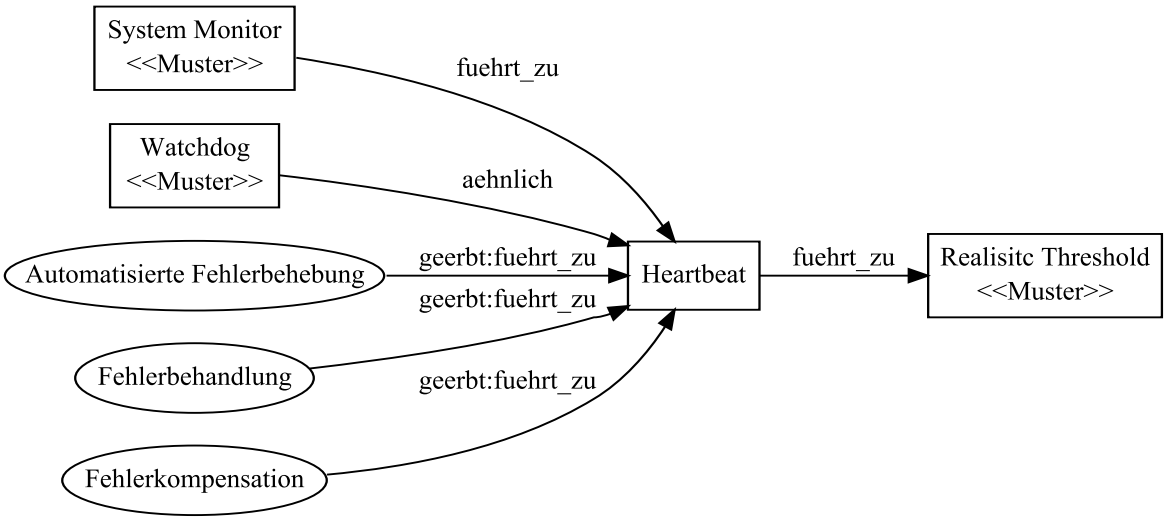
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Heartbeat" beschreibt den Ansatz, dass Komponenten deren Verfügbarkeit überwacht werden muss, in regelmäßigen Abständen Lebenszeichen an das überwachende System senden.

Beziehungen - Grafik




Beziehungen - Detail

- **System Monitor -> Heartbeat** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Watchdog -> Heartbeat** : Ähnliche Zielsetzung
  Quelle: Patterns for Fault tolerant Software; Hanmer; S. 107; Bei Heartbeat senden Komponenten Nachrichten über ihren Zustand an einen zentralen System Monitor. Bei

Watchdog werden die Komponenten von einer Überwachungskomponente aktiv überwacht. Die Watchdog Komponente platziert sich selbst zwischen den zu überwachenden Komponenten und dem System Monitor.

- **Heartbeat -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *92. I&A Requirements*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 192-206

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

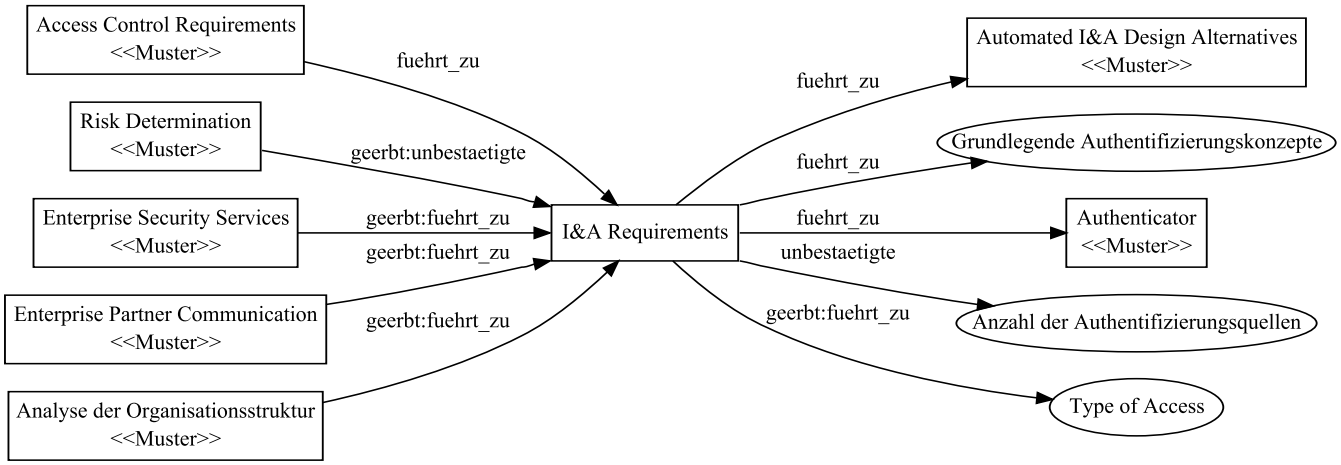
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "I&A Requirements" beschreibt die systematische Ermittlung und Dokumentation der Anforderungen an die Authentifizierung der mit dem System interagierenden Subjekte.

Beziehungen - Grafik




Beziehungen - Detail

- **Access Control Requirements -> I&A Requirements** : wenn Access Control dann auch Authentifizierung
  Wenn Access Control Mechanismen benutzt werden sollen, dann ist auch Authentifizierung notwendig. Aus diesem Grund müssen, als logische Konsequenz der Spezifikation von Access Control Requirements auch I&A Requirements spezifiziert werden.
- **I&A Requirements -> Automated I&A Design Alternatives** : dienen als Anforderungen für
  Siehe S. 207, Security Patterns, Schumacher et al.
- **I&A Requirements -> Grundlegende Authentifizierungskonzepte** : zur Architekturauswahl
  Die Ergebnisse der Anwendung des I&A-Requirements-Patterns bestimmen die Auswahl einer geeigneten I&A Lösung mittels des Automated-I&A-Design-Alternatives-Patterns, das Bestandteil der Gruppe grundlegende Authentifizierungskonzepte ist. Die Auswahl an Mechanismen erstreckt sich über eine Menge von Authentifizierungsverfahren (Benutzername/Passwort, Biometrie), zentrale- oder verteilte Verwaltung der Authentifizierungsinformation und der Verwendung von einem oder gleichzeitig mehreren Authentifizierungsverfahren.
- **I&A Requirements -> Authenticator** : Anforderungen an Authentifizierung
  Zur Umsetzung des Authenticator-Patterns werden Authentifizierungsanforderungen wie z. B. die Anzahl der Benutzer oder der Grad an benötigter Sicherheit vorausgesetzt. Siehe S. 325, Security Patterns, Schumacher et al.
- **I&A Requirements -> Anzahl der Authentifizierungsquellen** : Beeinflusst die Auswahl
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen

zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.

- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** :
  Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *93. Identity Federation*

Quellen

**Paper: A Pattern Language for Identity Management**

A Pattern Language for Identity Management; N. Delessy, E. Fernandez, M. Larrondo-Petrie; Proceedings of the International Multi-Conference on Computing in the Global Information Technology; 2007

Schwerpunkt(e)

- Sicherheit

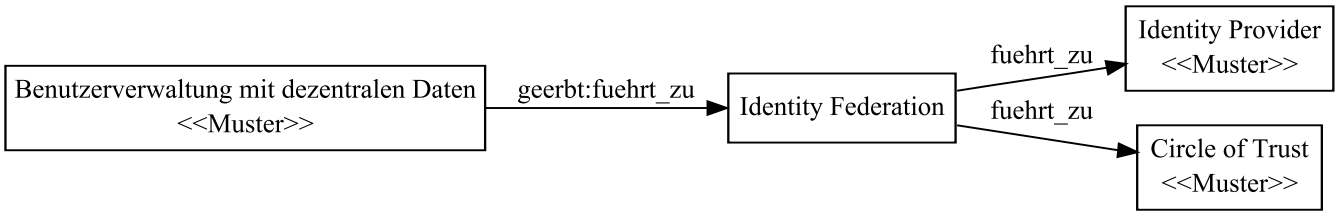
Gruppe(n)

- Identitätsmanagement

Kurzbeschreibung

Das Pattern "Identity Federation" beschreibt die gemeinsame Nutzung der Benutzer-Identität innerhalb eines Verbunds (Circle of Trust) von Dienste-Anbietern.

Beziehungen - Grafik




Beziehungen - Detail

- **Identity Federation -> Identity Provider** : beinhaltet
  Quelle: Abbildung 1 in A Pattern Language for Identity Management (Paper)
- **Identity Federation -> Circle of Trust** : benutzt
  Quelle: Abbildung 1 (Pattern diagram for identity management) in A Pattern Language for Identity Management (Paper)
- **Benutzerverwaltung mit dezentralen Daten -> Identitätsmanagement** : Zur Integration unterschiedlicher Benutzerstämme
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Pattern Benutzerverwaltung mit dezentralen Daten beschreibt einen Lösungsansatz der darauf basiert, dass jede Organisation ihre Benutzer selbst verwaltet und selbst in einem oder mehreren eigenen Systemen zur Benutzerverwaltung speichert. Das ist unter anderem der Fall, wenn verschiedene Primärsysteme mit jeweils eigener Benutzerverwaltung verwendet werden. Sollen derartige Systeme zu einem komplexen Systemverbund integriert werden, so ist die Verwendung von Patterns der Gruppe Identitätsmanagement zwangsläufig.

## *94. Identity Provider*

Quellen

**Paper: A Pattern Language for Identity Management**

A Pattern Language for Identity Management; N. Delessy, E. Fernandez, M. Larrondo-Petrie; Proceedings of the International Multi-Conference on Computing in the Global Information Technology; 2007

Schwerpunkt(e)

- Sicherheit

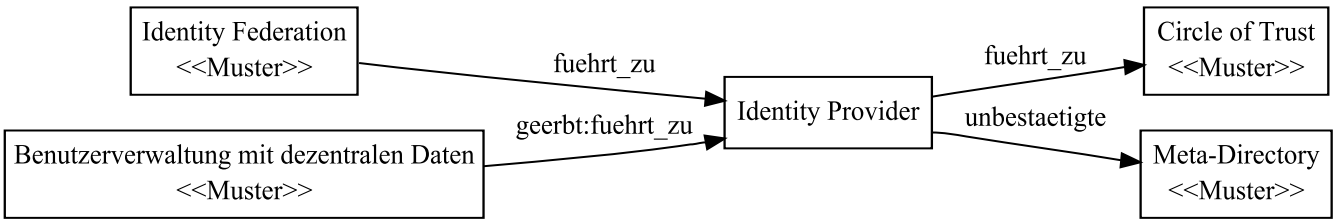
Gruppe(n)

- Identitätsmanagement

Kurzbeschreibung

Das Pattern "Identity Provider" beschreibt die Zentralisierung der Administration der Identitäts-Informationen zu Subjekten einer Sicherheits-Domäne.

Beziehungen - Grafik




Beziehungen - Detail

- **Identity Federation -> Identity Provider** : beinhaltet
  Quelle: Abbildung 1 in A Pattern Language for Identity Management (Paper)
- **Identity Provider -> Circle of Trust** : benutzt
  Abbildung 1 (Pattern diagram for identity management) in "A Pattern Language for Identity Management" zeigt die Beziehung.
- **Benutzerverwaltung mit dezentralen Daten -> Identitätsmanagement** : Zur Integration unterschiedlicher Benutzerstämme
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Pattern Benutzerverwaltung mit dezentralen Daten beschreibt einen Lösungsansatz der darauf basiert, dass jede Organisation ihre Benutzer selbst verwaltet und selbst in einem oder mehreren eigenen Systemen zur Benutzerverwaltung speichert. Das ist unter anderem der Fall, wenn verschiedene Primärsysteme mit jeweils eigener Benutzerverwaltung verwendet werden. Sollen derartige Systeme zu einem komplexen Systemverbund integriert werden, so ist die Verwendung von Patterns der Gruppe Identitätsmanagement zwangsläufig.

## 95. Individuals Decide Timing

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 174-176

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

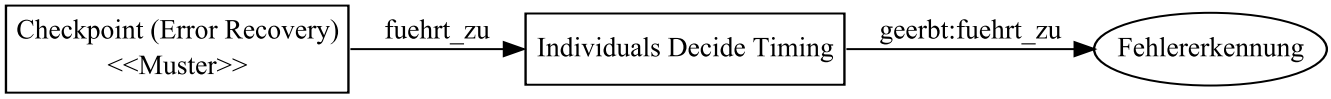
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Individuals Decide Timing" adressiert das Problem der Erstellung verteilter Snapshots.

Beziehungen - Grafik




Beziehungen - Detail

- **Checkpoint (Error recovery) -> Individuals Decide Timing** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## 96. Information Portal

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 6

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Anmerkung: Information Portal ist in der Quelle nicht in Patternform nach geeigneter Schablone beschrieben. Die Beschreibung lässt aber eine patternähnliche Verwendung zu.

### Schwerpunkt(e)

- Kommunikation

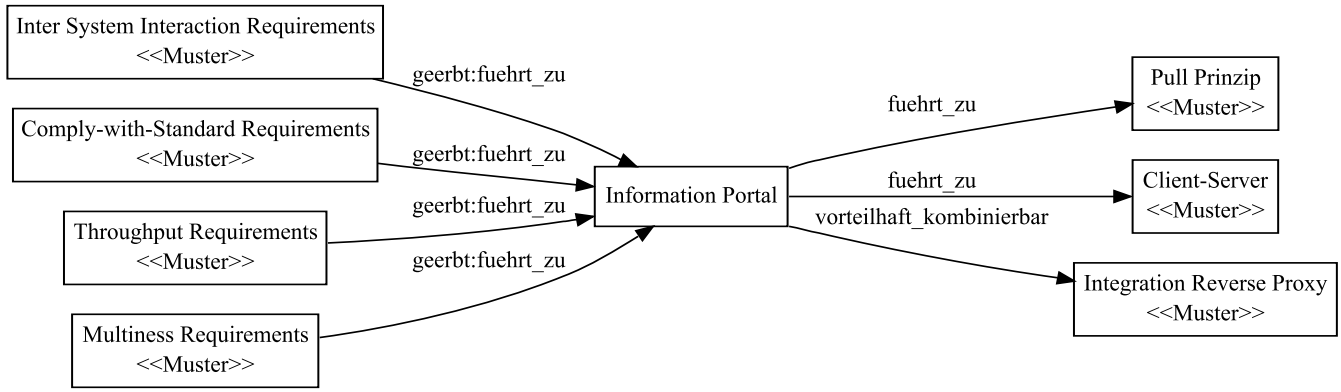
### Gruppe(n)

- Grundlegende Integrationsformen

### Kurzbeschreibung

Das Pattern "Information Portal" beschreibt eine Integrationsform, bei der die Integration zwischen Anwendungssystemen ausschließlich in der Benutzeroberfläche stattfindet.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Information Portal -> Pull Prinzip** : basiert auf Pull
  Das Information Portal holt Daten aktiv (Pull) aus den Quellsystemen um sie im Portal anzuzeigen.
- **Information Portal -> Client-Server** : eindeutige Rollenzuordnung
  Das Information Portal-Pattern beschreibt eine Systemarchitektur, in der der Zugriff auf eine Menge von Backend-Systemen (in der Rolle Server) über einen zentralen Zugriffspunkt (Information Portal) kanalisiert wird. In einer solchen Architektur bleibt das eindeutige Rollenverhältnis zwischen Client und Server erhalten.
- **Information Portal -> Integration Reverse Proxy** : Kann abgesichert werden durch Die Abbildung auf S. 468 in Security Patterns; Schumacher at al.; legt nahe, dass das Konzept des Information Portal durch die Anwendung von Integration Reverse Proxy und Demilitarized Zone auf eine geeignete sicherheitstechnische Basis gestellt werden kann.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung

Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform

- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.
- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## *97. Installability (and distribution) Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 274-279

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Flexibilität

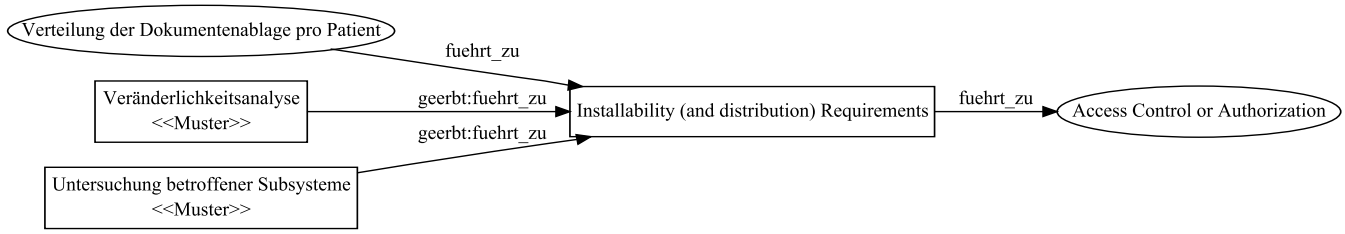
Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

Kurzbeschreibung

Das Pattern "Installability (and distribution) Requirements" beschreibt ein Verfahren zur Ermittlung der Anforderungen an die Einfachheit der Installation und die Verteilbarkeit der Software.

Beziehungen - Grafik




Beziehungen - Detail

- **Verteilung der Dokumentenablage pro Patient -> Installability (and distribution) Requirements** : Anforderungen zur verteilten Installation
  Je stärker die Dokumentenablage verteilt werden soll, desto wichtiger wird es, dass die Softwaresysteme der Knoten auf denen die Dokumentenablage erfolgt, einfach zu installieren ist.
- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *98. Integration Reverse Proxy*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 465-472

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

### Schwerpunkt(e)

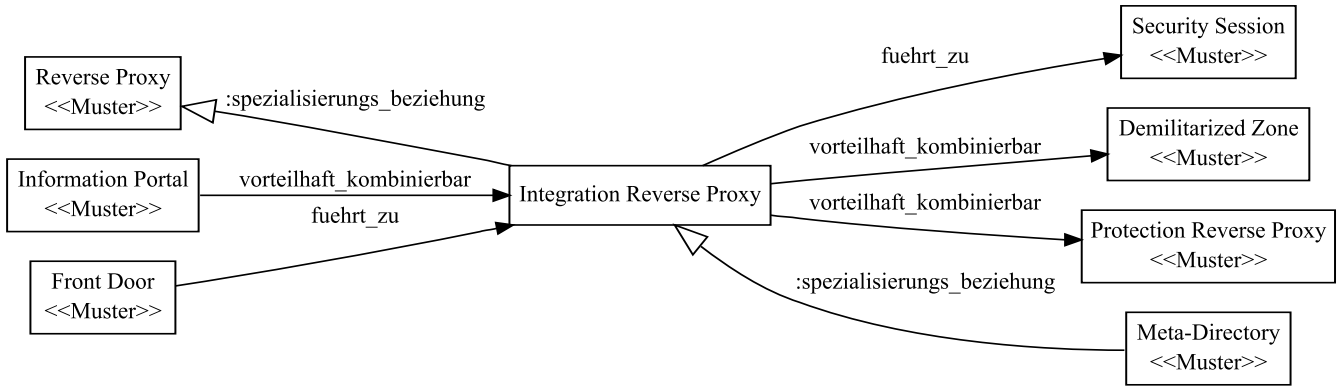
- Flexibilität

### Gruppe(n)

### Kurzbeschreibung

Das Pattern "Integration Reverse Proxy" beschreibt eine Form des Reverse Proxy, die dazu dient, so als Stellvertreter für mehrere nachgelagerte Server aufzutreten, dass der Eindruck entsteht, mit einem einzelnen Server zu kommunizieren.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Reverse Proxy -> Integration Reverse Proxy** : Ist eine Spezialform von
  Der Integration Reverse Proxy ist ein spezieller Reverse Proxy, der als Stellvertreter für mehrere Systeme agiert.
- **Information Portal -> Integration Reverse Proxy** : Kann abgesichert werden durch
  Die Abbildung auf S. 468 in Security Patterns; Schumacher at al.; legt nahe, dass das Konzept des Information Portal durch die Anwendung von Integration Reverse Proxy und Demilitarized Zone auf eine geeignete sicherheitstechnische Basis gestellt werden kann.
- **Front Door -> Integration Reverse Proxy** : Bestandteil
  Der Integration Reverse Proxy ist ein Bestandteil des Front-Door-Patterns. Front Door kann aber auch als Erweiterung des Integration-Reverse-Proxy-Patterns versanden werden.
  Quelle: S. 472; Security Patterns; Schumacher et al.
- **Integration Reverse Proxy -> Security Session** : relies on
  Quelle: Security Patterns; Schumacher et al.; S. 304
- **Integration Reverse Proxy -> Demilitarized Zone** : Hilfreiche Rahmenbedingung
  Quelle: S. 467 u. 468; Abbildung „Protection using an Integration Reverse Proxy“;

Security Patterns; Schumacher et al.
Eine Menge von Servern kann durch einen Integration Reverse Proxy in einer Demilitarized Zone (DMZ) gesichert werden.

- **Integration Reverse Proxy -> Protection Reverse Proxy** : Integration Protection Reverse Proxy
  Quelle: S. 470; Security Patterns; Schumacher et al.
  *"**Variants***
  *Integration Protection Reverse Proxy. It is easy (and wise) to combine the Integration Reverse Proxy with the Protection Reverse Proxy and gain the benefits of both."*
- **Integration Reverse Proxy -> Meta-Directory** :
  Ein Meta-Directory ist ein Integration Reverse Proxy für den transparenten Zugriff auf mehrere Benutzerdatenquellen.

## *99. Inter System Interaction Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 62

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation

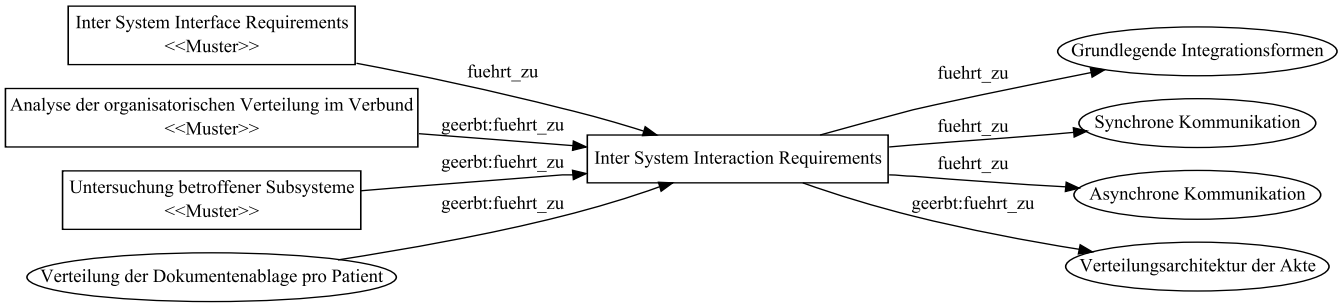
Gruppe(n)

- Kommunikationsanforderungen

Kurzbeschreibung

Das Pattern "Inter System Interaction Requirements" beschreibt ein Verfahren zur Ermittlung und Dokumentation der Anforderungen an die Art der Interaktion, die über eine Schnittstelle stattfindet oder stattfinden soll.

Beziehungen - Grafik



Beziehungen - Detail

- **Inter System Interface Requirements -> Inter System Interaction Requirements** : Vorbedingung
  Das Inter-System-Interaction-Requirements-Pattern verwendet die durch die Anwendung des Inter-System-Interface-Requirements-Patterns ermittelten Anforderungen und spezifiziert diese genauer.
  Quelle: Software Requirement Patterns; Withall; S. 63
  *"Use the inter-system interaction requirement pattern to specify a particular type of interaction across an inter-system interface."*
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Inter System Interaction Requirements -> Synchrone Kommunikation** : Auswahl
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.
- **Inter System Interaction Requirements -> Asynchrone Kommunikation** : Auswahl
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *100. Inter System Interface Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 51

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation

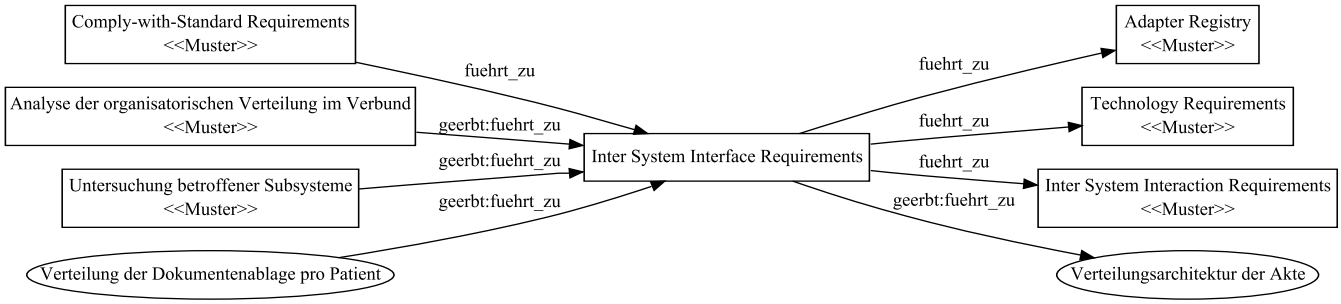
Gruppe(n)

- Kommunikationsanforderungen

Kurzbeschreibung

Das Pattern "Inter System Interface Requirements" dient der Ermittlung und Dokumentation der Schnittstellen (in Form von Anforderungen), die ein System oder eine Komponente zu anderen Systemen oder Komponenten besitzen soll.

Beziehungen - Grafik




Beziehungen - Detail

- **Comply-with-Standard Requirements -> Inter System Interface Requirements** : beeinflusst
  Quelle: Siehe S. 51, Punkt Related patterns; in Software Requirement Patterns; Withall
  Die Beschreibung zum Inter-System-Interface-Requirement-Pattern beschreibt das Comply-with-Standard-Requirement-Pattern als abhängiges Pattern.
- **Inter System Interface Requirements -> Adapter Registry** : bei vielzahl ähnlicher Subsysteme die gemeinsam genutzt werden sollen
- **Inter System Interface Requirements -> Technology Requirements** : Spezifikation der verwendeten Technologie
  Das Inter-System-Interface-Requirements-Pattern dient der Definition von Schnittstellen in Form von Anforderungen. Die beschriebenen Schnittstellen können durch Hinzufügen von Technology Requirements um die Spezifikation der zur

Umsetzung zu verwendenden Technologien erweitert werden.
Quelle: Software Requirement Patterns;Withall; S. 55
*"7. Technology to be used for the interface (if relevant)"*

- **Inter System Interface Requirements -> Inter System Interaction Requirements** : Vorbedingung
  Das Inter-System-Interaction-Requirements-Pattern verwendet die durch die Anwendung des Inter-System-Interface-Requirements-Patterns ermittelten Anforderungen und spezifiziert diese genauer.
  Quelle: Software Requirement Patterns; Withall; S. 63
  *"Use the inter-system interaction requirement pattern to specify a particular type of interaction across an inter-system interface."*
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *101. Interpreter*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 243-255

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

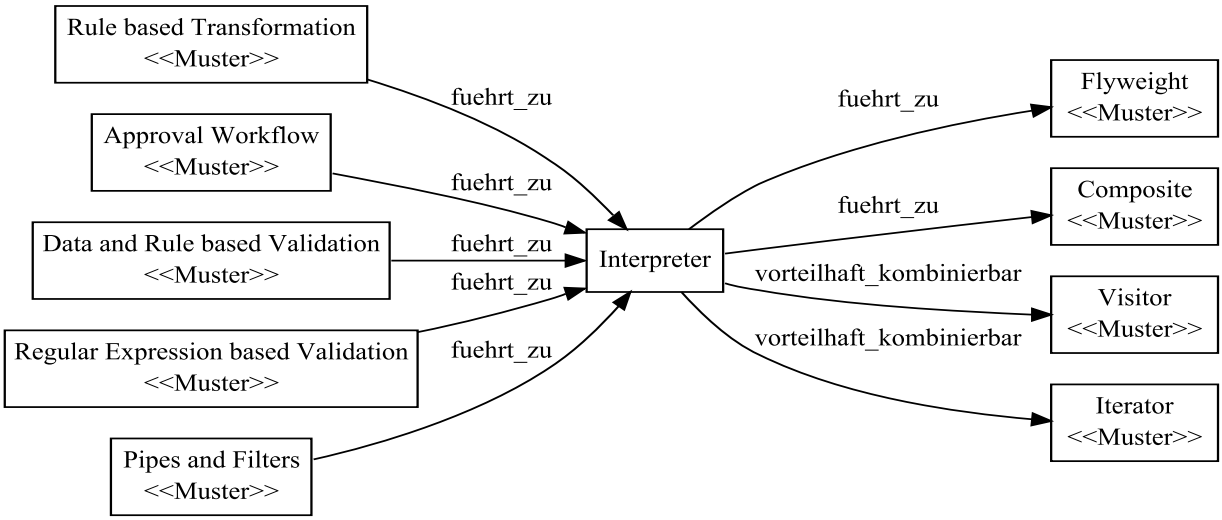
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Pattern "Interpreter" beschreibt generell den Ansatz, wiederkehrende Probleme, die einen hohen Grad an Flexibilität benötigen mittels dem Problem angepasster (evtl. kleiner) Sprachen zu lösen, die zur Laufzeit durch den Interpreter ausgeführt und nicht in Maschinencode überführt werden. Die Ausdrücke der Sprache beschreiben dann jeweils einen dem konkreten Problem angepassten Lösungsweg.

Beziehungen - Grafik




Beziehungen - Detail

- **Rule based Transformation -> Interpreter** : Interpretation der Regeln
  Zur Interpretation von Transformationsregeln wird ein Interpreter benötigt.
- **Approval Workflow -> Interpreter** : Interpretation der Workflow-Regeln
  Das Approval-Workflow-Pattern besagt, dass der Ablauf eines Approval Workflow durch eine, aus formalen Regeln bestehende Ablaufdefinition zu spezifizieren ist. Um diese Regeln in einem konkreten System anzuwenden, ist ein geeigneter Interpreter notwendig. Der Interpreter liest die Regeln und bringt sie anschließend zur Ausführung.
- **Kandidat: Data and Rule based Validation -> Interpreter** : Interpretation der Regeln
  Zur Interpretation der in Strings formulierten Regeln ist ein Interpreter notwendig. Dieser interpretiert die Regeln und wendet sie auf die zu überprüfenden Daten an. Gegebenenfalls werden zur Validierung (z. B. von Fremdschlüsselbeziehungen) aufgrund des Regel-Inhalts auch dritte Daten herangezogen.
- **Kandidat: Regular Expression based Validation -> Interpreter** : Interpretation der Regel
  Um einen regulären Ausdruck auf eine Zeichenkette anzuwenden, wird ein Interpreter

benötigt, der den regulären Ausdruck einliest und somit auf den String anwendbar macht.

- **Pipes and Filters -> Interpreter** :
  Quelle: POSA1; S. 57;
  S. 57 zeigt, dass bei der typischen Anwendung der Pipes-and-Filters-Architektur Interpreter und Bestandteile von Interpretern verwendet werden.
- **Interpreter -> Flyweight** : sharing terminal symbols
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Interpreter -> Composite** : defining grammar
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Interpreter -> Visitor** : adding operation
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma; S. 255; Related Patterns;
  *"Visitor can be used to maintain the behavior in each node in the abstract syntax tree in one class."*
- **Interpreter -> Iterator** :
  Quelle: Design Patterns; Gamma et al.; S. 255; Related Patterns;
  *"Iterator: The interpreter can use an Iterator to traverse the structure"*

## *102. Intrusion Detection Requirements*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 388-395

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

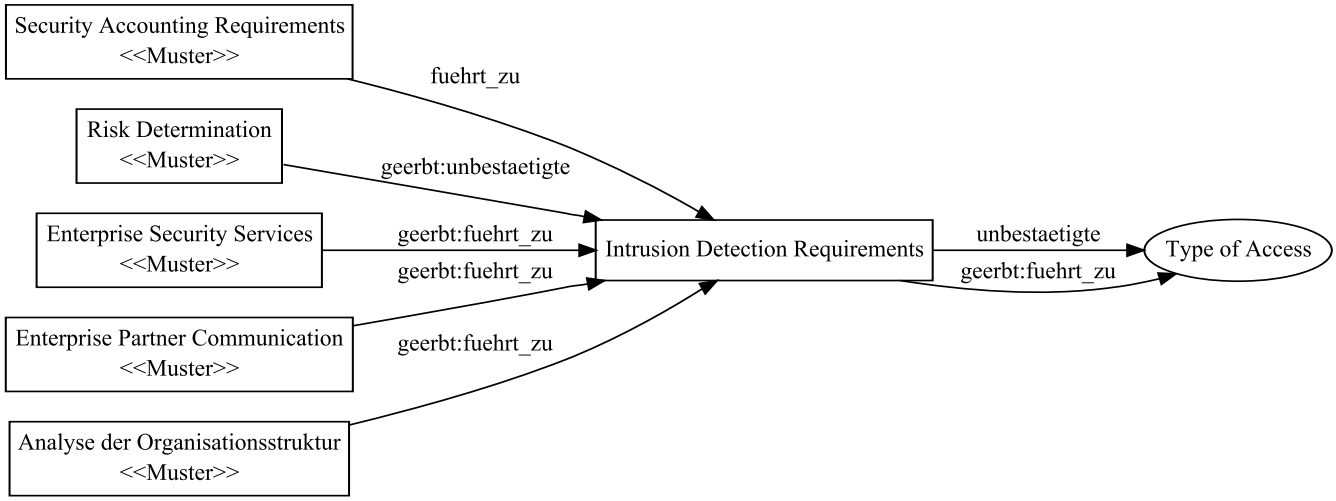
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "Intrusion Detection Requirements" beschreibt die systematische Identifikation und Dokumentation der Anforderungen an Mechanismen der Einbruchserkennung.

Beziehungen - Grafik




Beziehungen - Detail

- **Security Accounting Requirements -> Intrusion Detection Requirements** : Einbruchserkennung
  Die Anforderungsdefinition für eine Einbruchserkennung setzt die Kenntnis der Daten über erfasste sicherheitsrelevante Vorfälle voraus.
  Quelle: Security Patterns; Schumacher et al.; S. 388; Context
- **Intrusion Detection Requirements -> Type of Access** : beeinflusst vermutlich
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.
- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** : Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen

Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.

- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *103. Iterator*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 257-271

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

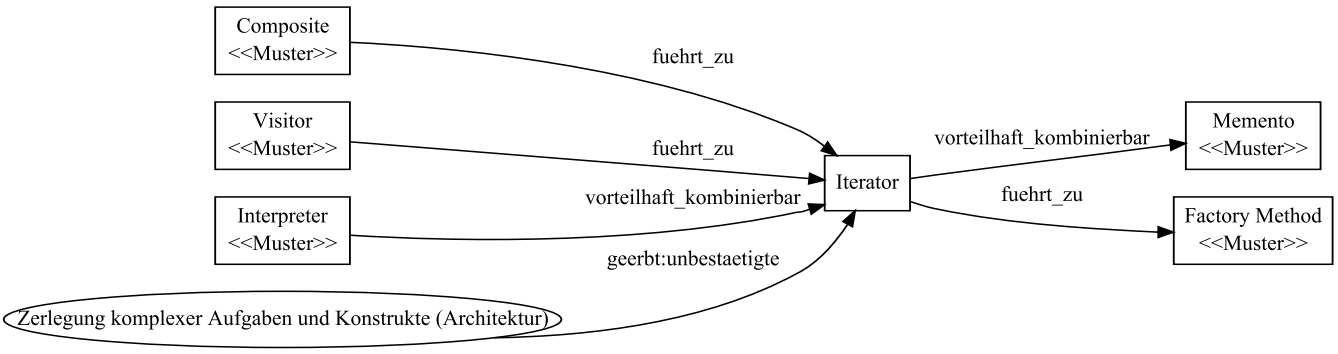
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

## Kurzbeschreibung

Die Idee des Iterator-Patterns besteht darin, für beliebige Objekt-Mengen einen Mechanismus der Iteration zur Verfügung zu stellen, der unabhängig von der tatsächlichen Abbildung in den zugrunde liegenden Datenstrukturen (wie z. B. Array, List, Set oder Map) ist.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Composite -> Iterator** : enumerating children
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Visitor -> Iterator** : defining traversals
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Interpreter -> Iterator** :
  Quelle: Design Patterns; Gamma et al.; S. 255; Related Patterns;
  *"Iterator: The interpreter can use an Iterator to traverse the structure"*
- **Iterator -> Memento** : Zur Speicherung des Iterationsstatus
  Quelle: Design Patterns; Gamma et al.; S. 271
  *"Memento is often used in conjunction with the Iterator pattern. An iterator can use a memento to capture the state of an iteratin. The iterator stores the memento internally."*
- **Iterator -> Factory Method** : Instanzierung geeigneter Iterator-Subklassen
  Quelle: Design Patterns; Gamma et al.; S. 271
  *"Factory Method (107): Polymorphic iterators rely on factory methods to instantiate the appropriate Iterator subclass."*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *104. Komponentenbasierte Softwarearchitektur*

Englische Bezeichnung

- Component based Software Architecture

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Grundlegende Architekturstile
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kontext

Es gibt verschiedene grundlegende Stilrichtungen oder Prinzipien in der Software-Architektur. Zu jeder dieser Stilrichtungen gibt es verschiedene Patterns, aber auch die Stilrichtung selbst lässt sich als Pattern beschreiben. Garlan bezeichnet die komponentenbasierte Softwarearchitektur in Quelle 1 als einen bekannten Architekturstil.

Problem
Die Wiederverwendung von bereits entwickelten Softwarebestandteilen kann helfen, den Aufwand für die Entwicklung eines Softwaresystems zu reduzieren. Aber nicht jedes Stück Software kann effizient wiederverwendet werden. Es muss zur Wiederverwendung geeignet sein.

Lösung
Es gibt verschiedene Kriterien, die die Wiederverwendbarkeit von Software verbessern. Bei wiederverwendbaren Softwarebestandteilen spricht man von Software-Komponenten. Die Eigenschaften von Softwarekomponenten werden in der vorliegenden Arbeit bereits im Kapitel 2.1.6. Software-Komponente umfassend beschrieben.

Eine komponentenbasierte Softwarearchitektur verfolgt als zentralen Ansatz die Gliederung der Problemdomäne in einzelne, Softwarekomponenten zuordenbare Aufgabenbereiche. Die so entwickelten Komponenten haben einen klar abgegrenzten Aufgabenbereich und können dadurch eindeutig benannt werden. Die eindeutige Benennung wiederum ermöglicht, gemeinsam mit verständlicher Dokumentation, Kapselung der Komponentenschnittstelle und dank einer geeigneten Schnittstelle die Wiederverwendung der Komponente.

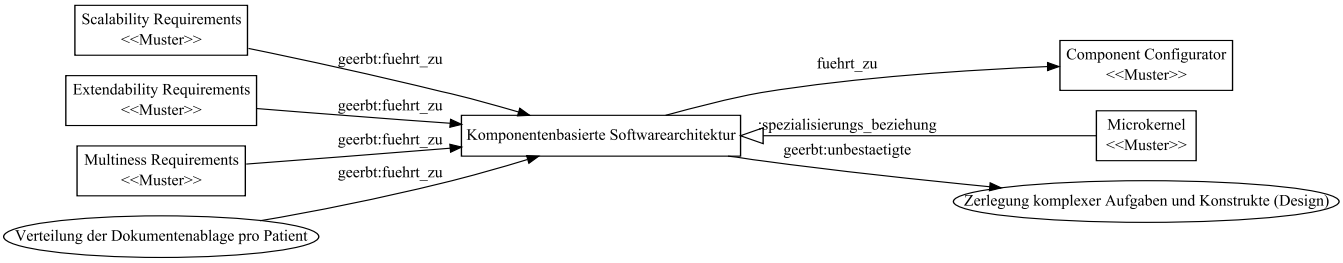
Quellen

1. David Garlan, and Mary Shaw, In V. Ambriola and G. Tortora (ed.), An Introduction to Software Architecture, Advances in Software Engineering and Knowledge Engineering, Series on Software Engineering and Knowledge Engineering, Vol 2, World Scientific Publishing Company, Singapore, pp. 1-39, 1993. Also available as: Carnegie Mellon University Technical Report CMU-CS-94-166, January 1994.

Online verfügbar unter:
http://www.cs.cmu.edu/afs/cs/project/able/ftp/intro_softarch/intro_softarch.pdf

2. G.T. Heineman und W.T. Councill, Component-Based Software Engineering: Putting the Pieces Together, Addison-Wesley Longman, Amsterdam, 2001.
3. M.D. McIlroy, Mass Produced Software Components, Software Engineering, Report on a conference sponsored by the NATO Science Committee, Garmisch: 1968, S. 138-155.
4. J. Sametinger, Software engineering with reusable components, Berlin [u.a.]: Springer, 1997.

Beziehungen - Grafik




Beziehungen - Detail

- **Komponentenbasierte Softwarearchitektur -> Component Configurator** : benötigt als Bestandteil
  Um eine komponentenbasierte Architektur umsetzen zu können, in der Komponenten dynamisch konfigurierbar sind, wird zur Umsetzung das Component-Configurator-Pattern benötigt.
- **Komponentenbasierte Softwarearchitektur -> Microkernel** : Spezialform
  Die Microkernelarchitektur ist eine Architekturform, bei der ein kleiner Kernbestandteil durch hinzufügen von Komponenten zu einem komplexen und vielfältig funktionalen Gesamtsystem erweitert wird.
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.

- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.
- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *105. Leaky Bucket Counter*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 135-137

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

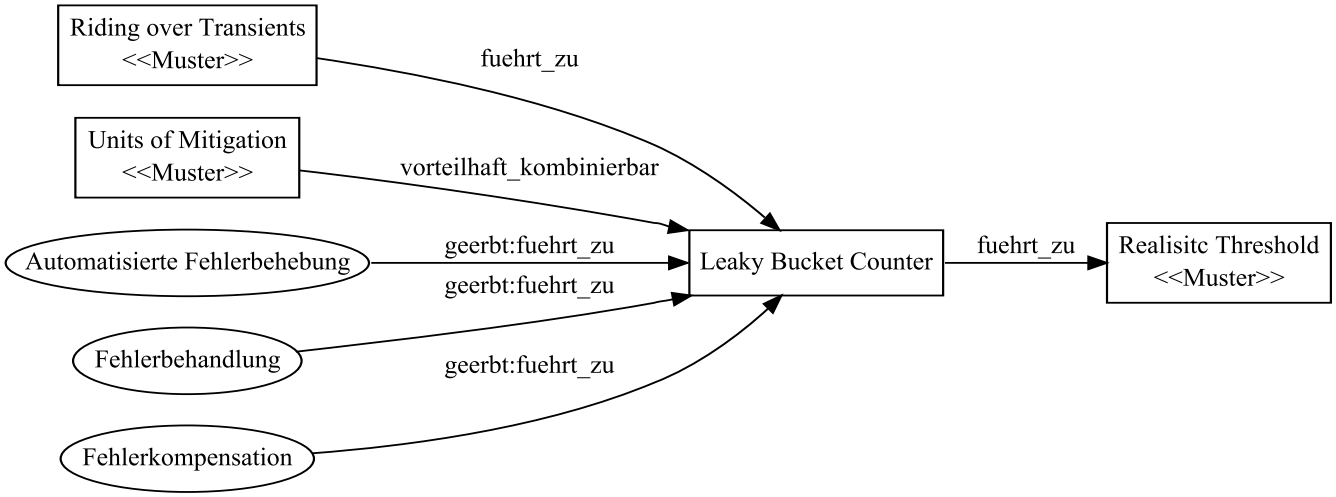
- Fehlererkennung

Kurzbeschreibung

Das Pattern "Leaky Bucket Counter" beschreibt ein Verfahren, das bei Fehlern mit geringer Auswirkung auf das Systemverhalten zum Einsatz kommen kann. Diese einzeln nicht

kritischen Fehler können sporadisch oder auch gehäuft auftreten. Der Leaky Bucket Counter dient der Erkennung von gehäuftem Auftreten solcher Fehler.

Beziehungen - Grafik




Beziehungen - Detail

- **Riding over Transients -> Leaky Bucket Counter** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Units of Mitigation -> Leaky Bucket Counter** : Überwachung ungewöhnlicher Fehlerhäufigkeiten
  Für jede Unit of Mitigation kann ein Leaky Bucket Counter angelegt werden. Der Leaky Bucket Counter dient dazu, bei ungewöhnlich häufigem Auftreten von Fehlern innerhalb kurzer Zeit einen Alarm auslösen zu können. Das kann z. B. durch Versenden einer Nachricht etc. erfolgen.
  Quelle: Pattern for Fault Tolerant Software; Hanmer; S. 136
- **Leaky Bucket Counter -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte

Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *106. Let Sleeping Dogs Lie*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 230-232

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

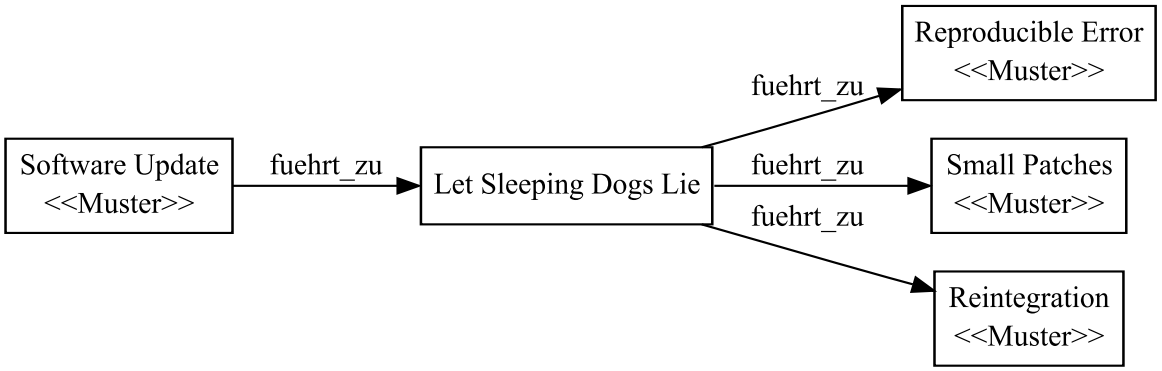
Gruppe(n)

- Fehlerkorrektur

Kurzbeschreibung

Das Pattern "Let Sleeping Dogs Lie" beschreibt die Überlegung, Fehler dann nicht zu korrigieren, wenn ihre Auswirkungen gering und das Risiko, durch die Fehlerkorrektur Folgefehler zu produzieren groß ist.

Beziehungen - Grafik




Beziehungen - Detail

- **Software Update -> Let Sleeping Dogs Lie** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Let Sleeping Dogs Lie -> Reproducible Error** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Let Sleeping Dogs Lie -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

- **Let Sleeping Dogs Lie -> Reintegration** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

## *107. Limit Retries*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 160-162

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

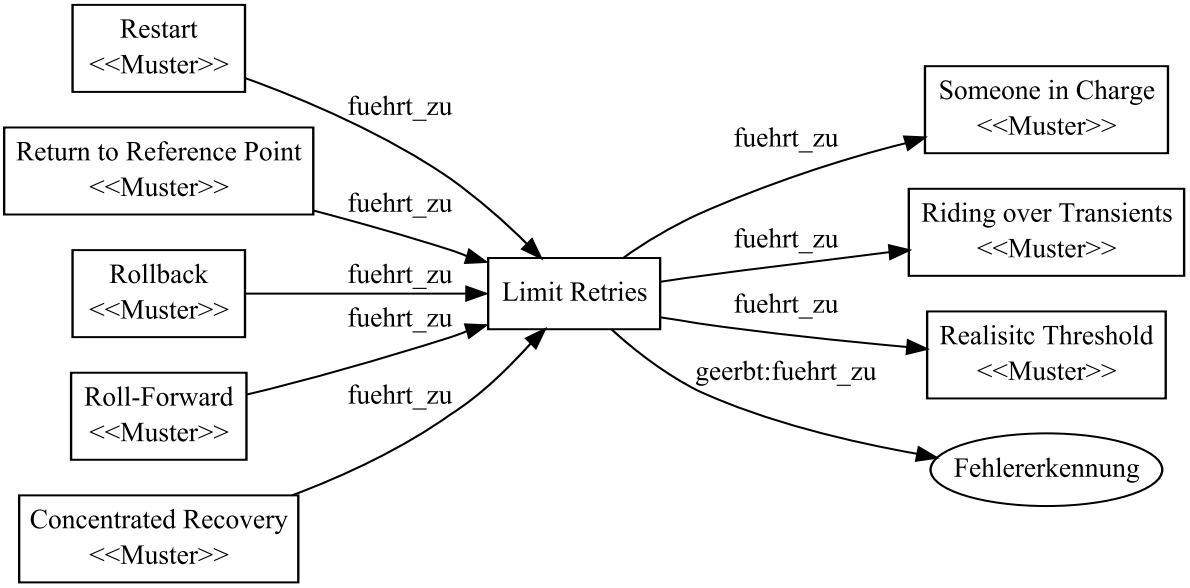
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Limit Retries" besagt, dass die Anzahl der Versuche, eine Anfrage zu verarbeiten nicht endlos sein darf. Das ist vor allem dann relevant, wenn Mechanismen der Fehlerkompensation eingesetzt werden, die auf dem nochmaligen Verarbeiten der Anfrage im Fehlerfall basieren (vgl. z. B. Failover oder Restart).

Beziehungen - Grafik




Beziehungen - Detail

- **Restart -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47

- **Return to Reference Point -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Rollback -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Roll-Forward -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Concentrated Recovery -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Limit Retries -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Limit Retries -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Limit Retries -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *108. Limited Access*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 312-319

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

Unter dem Namen Limited View zu finden auf S. 19

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

Schwerpunkt(e)

- Sicherheit

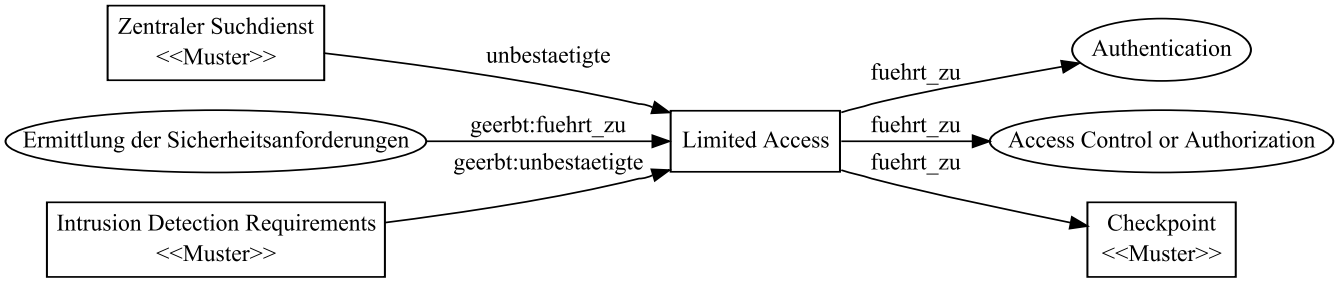
Gruppe(n)

- Type of Access

## Kurzbeschreibung

Das Pattern "Limited Access" beschreibt den Ansatz, einem Benutzer nur genau die Funktionen zur Verfügung zu stellen, zu deren Nutzung er berechtigt ist. Die Oberflächen-Elemente zu deren Nutzung keine Berechtigung vorliegt werden folglich nicht angezeigt.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Zentraler Suchdienst -> Limited Access** : Steuerung der Zugriffsberechtigung auf Suchergebnisse
  Die Verwendung eines zentralen Suchdienstes führt dazu, dass entweder eine Full-Access-with-Errors-Lösung in Verbindung mit dezentralen Autorisierungslösungen oder eine Limited-Access-Lösung mit zentralen Autorisierungslösungen denkbar ist.
- **Limited Access -> Authentication** : Ermittlung des Benutzers
  Um einen durch Limited Access beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Limited Access -> Checkpoint** :
  Quelle: Security Patterns; Schumacher et al.; S. 319
  *"Check Point* [...] *should be considered for designing and implementing the user I&A and association of access rights."*
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem

Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

- **Intrusion Detection Requirements -> Type of Access** : beeinflusst vermutlich
Geerbte Beziehung: unbestaetigte_beziehung

## *109. Lokale Autorisierungsregeln*

Englische Bezeichnung

- Local Storage and Management of Autorization Rules

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Verteilung der Autorisierungsinformation

Kontext

In einem System mit verteilter Ablage von Patientendaten müssen Autorisierungsregeln verwaltet werden.

Problem
Die beteiligten Subsysteme oder mindestens einige der beteiligten Subsysteme können ausschließlich in ihnen selbst gespeicherte Autorisierungsregeln, Rollenkonzept usw. verarbeiten.

Lösung
Jedes Subsystem, das Patientendaten beinhaltet, verwaltet die Autorisierungsregeln für den Zugriff auf seine Daten selbst.

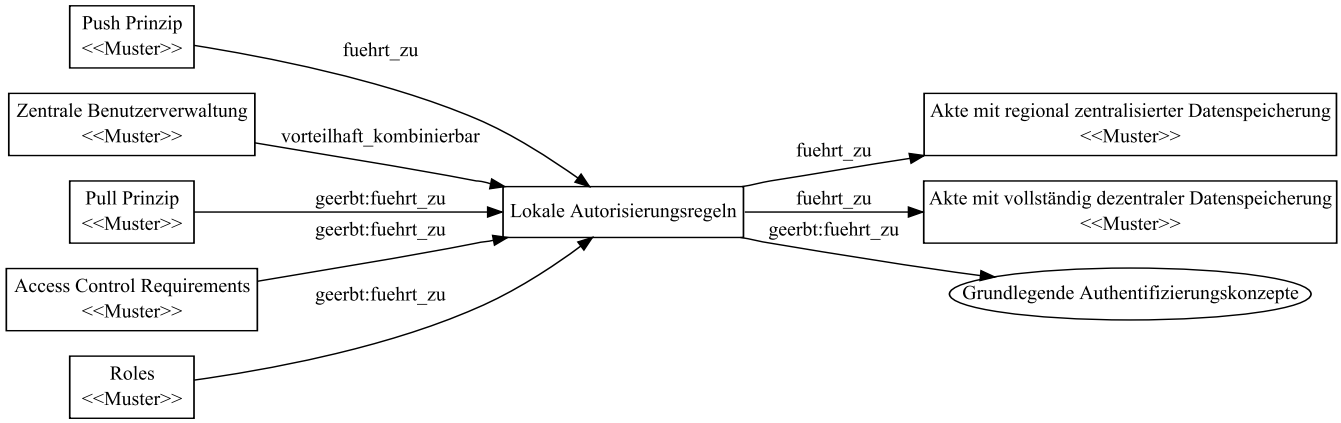
Vorteile:

- An den bestehenden Systemen muss nichts verändert werden

Probleme:

- Die Berechtigungen eines Nutzers zu einem Zeitpunkt sind über mehrere Subsysteme hinweg kaum zu überblicken.
- Das Risiko von fehlerhafter Berechtigungsvergabe besteht durch die resultierende große Zahl an Berechtigungsverwaltern.

Beziehungen - Grafik




Beziehungen - Detail

- **Push Prinzip -> Lokale Autorisierungsregeln** : zwangsläufig
  Werden die Daten nach dem Push-Prinzip an die verschiedenen (jeweils eigenständigen) Systemknoten verteilt, kann der Zugriff auf diese Daten auch ausschließlich über deren lokale Autorisierungsregeln gesteuert werden.
- **Zentrale Benutzerverwaltung -> Lokale Autorisierungsregeln** .
  Eine zentrale Benutzerverwaltung kann auch mit lokalen Autorisierungsregeln, also Autorisierungsregeln die z. B. pro Knoten einer verteilten Krankenakte verwaltet werden, kombiniert werden. Vorteil bei dieser Kombination ist, dass für alle im Gesamtsystem verteilt existierenden Berechtigungen Regeln auf Basis von eindeutigen Benutzern existieren.
- **Lokale Autorisierungsregeln -> Akte mit regional zentralisierter Datenspeicherung** : kombinierbar
  Sollen lokale Autorisierungsregeln verwendet werden, so sind diese mit Akten mit regional zentralisierter Datenspeicherung kombinierbar. Es gilt allerdings die Einschränkung, dass sich in diesem Fall der Begriff lokal ausschließlich auf die Daten verwaltenden Knoten bezieht. Diese sind bei einer Akte mit regional zentralisierter Datenspeicherung allerdings ausschließlich in den regionalen Zentren verfügbar.
- **Lokale Autorisierungsregeln -> Akte mit vollständig dezentraler Datenspeicherung** : kombinierbar
  Lokale Autorisierungsregeln können in einer Akte mit vollständig dezentraler Datenspeicherung umgesetzt werden. Dann besitzt jeder Knoten seine eigenen Zugriffsregeln, die den Zugriff auf die in ihm enthaltenen Objekte (med. Dokumente) regeln.
- **Pull Prinzip -> Verteilung der Autorisierungsinformation** : Gewährleistung des Zugriffsschutzes
  Geerbte Beziehung: fuehrt_zu_beziehung
  Erfolgt der Transport der Akteninhalte nach dem Pull-Prinzip, so ist die verteilte oder zentrale Verwaltung von Autorisierungsregeln zwangsläufig notwendig um beim Zugriff den Schutz vor unberechtigtem Zugriff zu gewährleisten.
- **Access Control Requirements -> Verteilung der Autorisierungsinformation** : aktenbezogene Auswahl des Autorisierungsmechnismus

Geerbte Beziehung: fuehrt_zu_beziehung
Die spezifizierten Access Control Requirements dienen als Informationsbasis für die Auswahl eines geeigneten Autorisierungskonzepts für die verteilte Krankenakte.

- **Roles -> Aktenbezogene Autorisierungskonzepte** : Organisationsbezug
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Roles-Pattern beschreibt die Ermittlung und Definition von Rollen und zugehörigen Berechtigungen für Benutzer eines Systems. Im komplexen Umfeld einer verteilten Krankenakte haben Benutzer diese Rollen aber nur bezogen auf bestimmte Organisationen oder Organisationseinheiten. Bei der Umsetzung aktenbezogener Autorisierungskonzepte mit zentral verwalteten Autorisierungsregeln ist deshalb auch die Berücksichtigung des Organisationsbezugs von Rollen und Rechten zu bedenken.
- **Verteilung der Autorisierungsinformation -> Grundlegende Authentifizierungskonzepte** : Benutzeridentifikation
  Geerbte Beziehung: fuehrt_zu_beziehung.
  Um die Gültigkeit von Autorisierungsbedingungen bei einem Funktionsaufruf prüfen zu können, ist die Identifikation des zu Berechtigenden mittels Authentifizierung notwendig.

## *110. Lookup*

Quellen

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 495-496

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Kommunikation
- Flexibilität

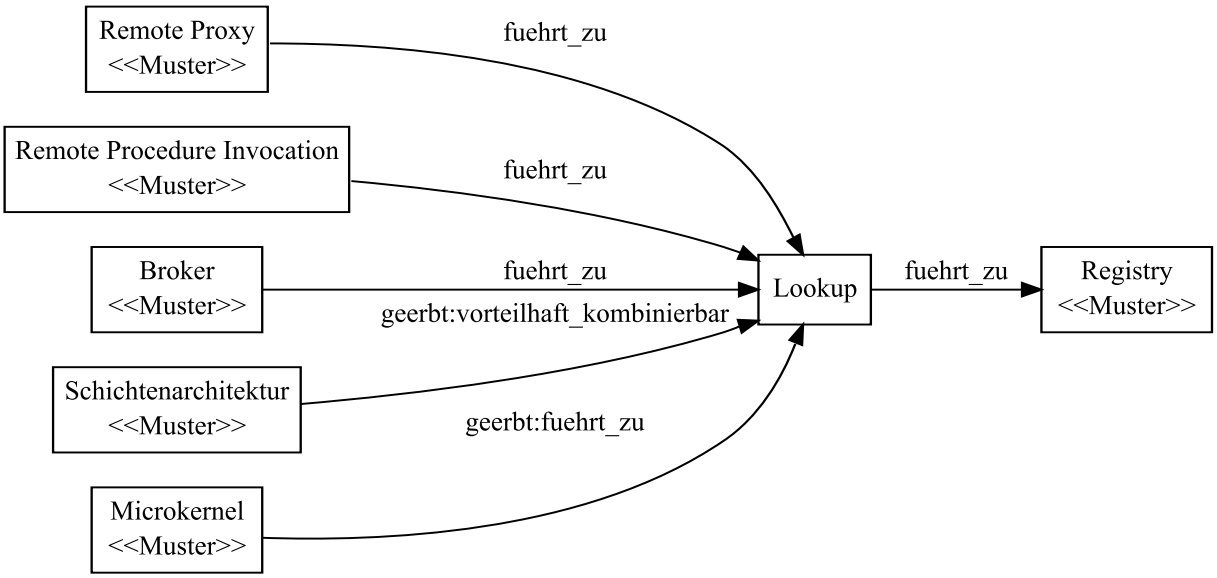
Gruppe(n)

- Kopplung und Entkopplung
- Verbergen der Kommunikation

Kurzbeschreibung

Das Lookup-Pattern beschreibt ein Verfahren, in dem (entfernte) Komponenten mittels symbolischer Namen adressiert werden. Die Verwendung des symbolischen Namens macht es möglich, die Komponente zum Zeitpunkt der Codierung zu adressieren ohne deren tatsächliche, zum Zeitpunkt des Betriebs bestehende Adresse zu kennen.

Beziehungen - Grafik




Beziehungen - Detail

- **Remote Proxy -> Lookup** : Instanzierung
  Remote Proxy benötigt Lookup oder Factory zur Instanzierung wenn nicht bei jeder Instanzierung vollständige Verbindungsdaten zur entfernten Implementierung als Parameter übergeben werden sollen.
- **Remote Procedure Invocation -> Lookup** : Bestandteil wenn symbolische Namen verwendet werden
  Wenn symbolische Namen verwendet werden ist Lookup ein Bestandteil der Remote Procedure Invocation.
- **Broker -> Lookup** : kann Lookup Funktionalität anbieten
  Lt. Pattern Oriented Software Architecture, Volume 4, S. 239 kann eine Broker-Implementierung zusätzlich Lookup-Funktionalität anbieten, so dass Clients ortstransparent zugreifen können.
- **Lookup -> Registry** : Registry als Datenquelle
  Eine Registry ist eine mögliche Datenquelle für die Durchführung eines Lookup.
  Beispiel: RMIRegistry in Java
  (http://download.oracle.com/javase/6/docs/api/java/rmi/registry/Registry.html)
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung,

wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *111. Maintenance Interface*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 63-65

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

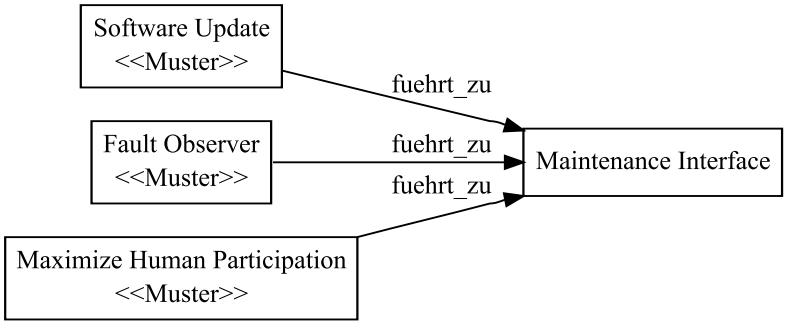
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Maintenance Interface" befasst sich mit der Frage, ob die für Wartungszugriffe notwendige Interaktion über die Standard-Mittel eines Systems oder über separate Wartungswege erfolgen soll.

Beziehungen - Grafik




Beziehungen - Detail

- **Software Update -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Fault Observer -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Maximize Human Participation -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35

## *112. Marked Data*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 220-223

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

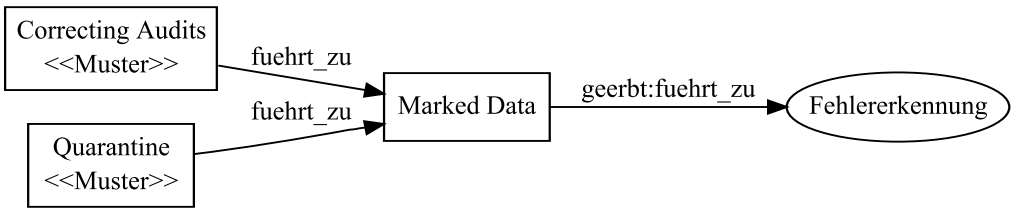
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Marked Data" befasst sich mit den Rahmenbedingungen, unter denen als fehlerhaft erkannte Daten als fehlerhaft markiert werden sollen, um sie vor weiterer Verarbeitung und somit der Verbreitung des Fehlers auszuschließen.

Beziehungen - Grafik




Beziehungen - Detail

- **Correcting Audits -> Marked Data** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Quarantine -> Marked Data** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung.
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## 113. Master Patient Index

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

- Patientenidentifikation

Kontext

Eine Menge von Subsystemen soll zu einer verteilten Krankenakte integriert werden.

Problem
Patienten haben in den verschiedenen Subsystemen der verteilten Krankenakte verschiedene Nummern. Deshalb ist ein Patient ohne zusätzliche Mechanismen nicht über die Subsystemgrenzen hinweg identifizierbar.

Lösung
An einer zentralen Stelle werden die Nummern aus den verschiedenen Einrichtungen und deren Subsystemen zueinander in Beziehung gesetzt. Dazu ist es notwendig, die IDs eines Patienten aus allen Subsystemen, in denen er verwaltet wird, einer Master-ID zuzuordnen. Diese ist dem Patienten eindeutig zugeordnet.

Über die daraus resultierende Datensammlung kann eine Patienten-ID aus einem der Subsysteme in die Patienten-ID desselben Patienten in einem beliebigen anderen Subsystem gemappt werden. Zur Ermittlung der Patienten-ID im Zielsystem ist die Angabe des Quellsystems und der Patienten-ID notwendig. Durch diese beiden Parameter lässt sich die Master-ID des Patienten ermitteln. Die Master-ID des Patienten kann wiederum verwendet werden, um die Patienten-ID im Zielsystem zu ermitteln.

Beispiele
**Beispiel:** Einfacher Master Patient Index
Patient: Max Mustermann; MasterID: 0123456789

- Subsystem: A; PatientID: 1357
- Subsystem: D; PatientID: 1357
- Subsystem: E; PatientID: 1357
- Subsystem: B; PatientID: X121523
- Subsystem: C; PatientID: RC1537766

**Beispiel:** Master Patient Index mit flexibler Zuordnung von Einrichtungen und Subsystemen.
Patient: Max Mustermann; MasterID: 0123456789

- PatientIndexID: 0001; PatientID: 1357
- PatientIndexID: 0002; PatientID: X121523

- PatientIndexID: 0003; PatientID: RC1537766

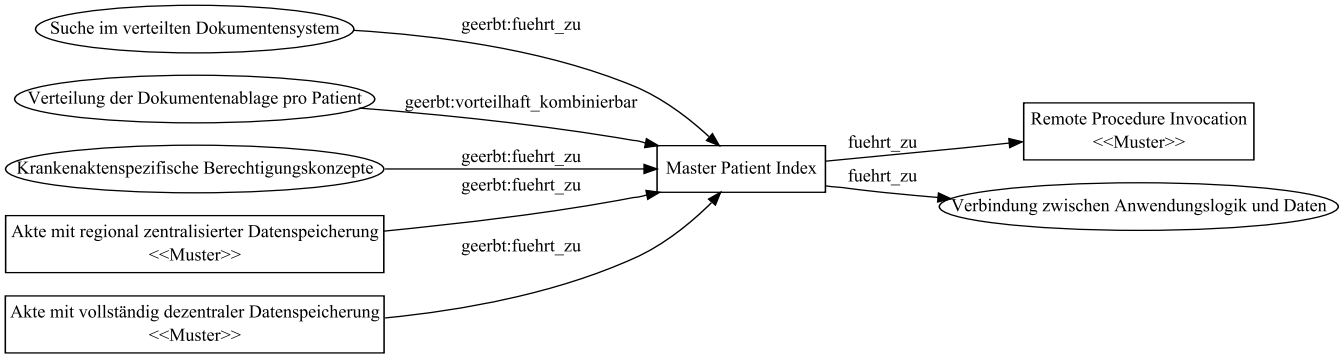
Subsystem-Zuordnung:

- Einrichtung: Krankenhaus A; EinrichtungID: 1; Subsystem: A; PatientIndexID: 0001
- Einrichtung: Krankenhaus A; EinrichtungID: 1; Subsystem: B; PatientIndexID: 0001
- Einrichtung: Krankenhaus A; EinrichtungID: 1; Subsystem: C; PatientIndexID: 0001
- Einrichtung: Arzt B; EinrichtungID: 2; Subsystem *; PatientIndexID: 0002
- Einrichtung: Reha-Klinik C; EinrichtungID: 3; Subsystem KIS; PatientIndexID: 0003

Quellen:


- Laurie A Rinehart-Thompson, “Storage Media Profiles and Health Record Retention Practice Patterns in Acute Care Hospitals,” Perspectives in Health Information Management / AHIMA, American Health Information Management Association 5.
- http://www.eh-cc.de/wp-content/html/agdgi/down/6_MPI-Brandner.pdf; Letzter Download (07.10.2011)


Beziehungen - Grafik




Beziehungen - Detail

- **Master Patient Index -> Remote Procedure Invocation** : Notwendig
  Um einen Master Patient Index umzusetzen ist es notwendig, dass dessen Schnittstelle von entfernten Softwaresystemen aufgerufen werden kann. Die entfernten Aufrufe dienen sowohl der Abfrage von IDs eines Patienten in anderen Systemen als auch der Registrierung von IDs für neue Patienten.
- **Master Patient Index -> Verbindung zwischen Anwendungslogik und Daten** : Speicherung des Index
  Die Daten des Master Patient Index müssen persistent gepeichert werden um dauerhaft verwendet werden zu können.
- **Suche im verteilten Dokumentensystem -> Patientenidentifikation** : Zur eindeutigen Zuordnung der Suchergebnisse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um in einem verteilten Dokumentensystem gespeicherte Dokumente zu einem Patienten zu suchen, ist es notwendig, dass der Patient über die Grenzen der einzelnen Subsystem-Knoten hinweg eindeutig identifizierbar bleibt. Besonders Systeme mit

heterogenen Patienten-IDs benötigen einen Mechanismus zur subsystemübergreifenden Patientenidentifikation.

- **Verteilung der Dokumentenablage pro Patient -> Patientenidentifikation** :
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wenn Dokumente verteilt abgelegt werden ist es hilfreich, wenn die zugehörigen Patienten über die verschiedenen Systemknoten hinweg eindeutig identifizierbar sind.
- **Krankenaktenspezifische Berechtigungskonzepte -> Patientenidentifikation** :
  Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im medizinischen Kontext ist die zentrale Entität i.d.R. der Patient. Seine eindeutige Identifizierung, auch über Systemgrenzen hinweg, ist die Basis der meisten medizinspezifischen Autorisierungsmechanismen.
- **Akte mit regional zentralisierter Datenspeicherung -> Patientenidentifikation** :
  falls Patienten nicht bereits eindeutig identifizierbar sind
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Akte mit vollständig dezentraler Datenspeicherung -> Patientenidentifikation** :
  falls Patienten nicht bereits eindeutig identifizierbar sind
  Geerbte Beziehung: fuehrt_zu_beziehung

## *114. Maximize Human Participation*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 60-62

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

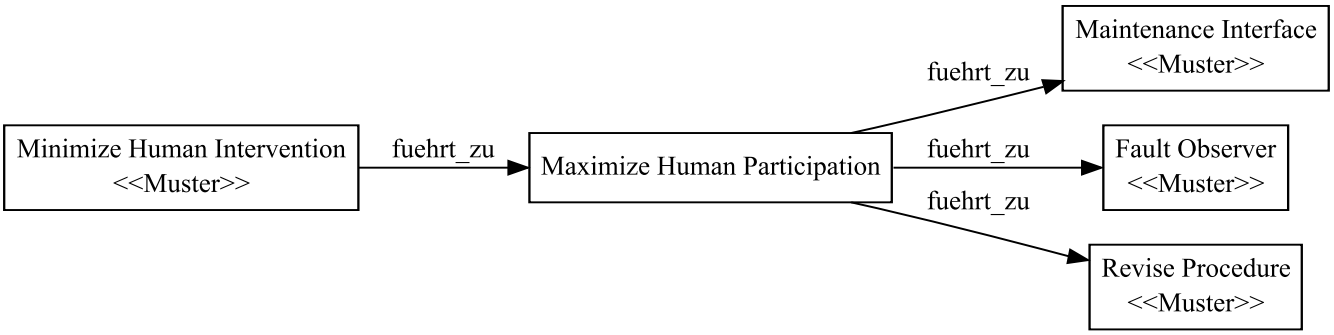
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Es gibt Situationen, in denen ein System nicht selbstständig entscheiden kann, wie ein Fehler behandelt werden soll. Mit diesen Situationen befasst sich das Pattern "Maximize Human Participation".

Beziehungen - Grafik




Beziehungen - Detail

- **Minimize Human Intervention -> Maximize Human Participation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Maximize Human Participation -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Maximize Human Participation -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Maximize Human Participation -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *115. Mediator*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 273-282

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

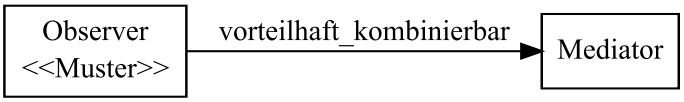
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Pattern "Mediator" beschreibt einen Ansatz, zueinander in Beziehung stehende Objekte über ein drittes Objekt, den sog. Mediator zu verbinden, um die Objekte nicht direkt zu koppeln.

Beziehungen - Grafik




Beziehungen - Detail

- **Observer -> Mediator** : Behandlung komplexer Abhängigkeiten zwischen Colleagues
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Quelle: Design Patterns; Gamma et al.; S. 282
  *„Colleagues can communicate with the mediator using the Observer pattern.“*

## *116. Mehrfaktorige Authentifizierung*

Englische Bezeichnung

- Two or More Factor based Authentication

Quellen

**IT-Sicherheit, Eckert**

S. 431 unten

**Mehr-Faktor-Authentifikation:**

*„Zur Realisierung von Authentifikationsverfahren werden häufig Kombinationen von ein oder mehr Authentifikationstechniken verwendet, um auf diese Weise die Sicherheit zu erhöhen und um die Vorteile der unterschiedlichen Techniken gezielt auszunutzen. Bei einer Kombination aus zwei Techniken spricht man deshalb von einer zwei Faktor-Authentifikation, die in der Regel darauf basiert, dass man etwas wissen und etwas besitzen muss. Ein allgemein geläufiges Beispiel dafür ist die Authentifikation eines Bankkunden an einem Geldautomat. Der Kunde beweist seine Identität zum einen durch den Besitz seiner ec-Karte und zum anderen durch die Kenntnis seiner PIN (spezifisches Wissen). Da sich in diesem Szenario nur der Kunde gegenüber dem Geldautomaten authentifiziert, sehen wir hier aber auch ein klassisches Beispiel für eine fehlende wechselseitige Authentifikation.“*

“10 Authentifikation,” in IT-Sicherheit, Bd. 5 (Oldenbourg Wissenschaftsverlag GmbH, 2011), 429-533, http://dx.doi.org/10.1524/9783486595970.429.

Schwerpunkt(e)

- Sicherheit

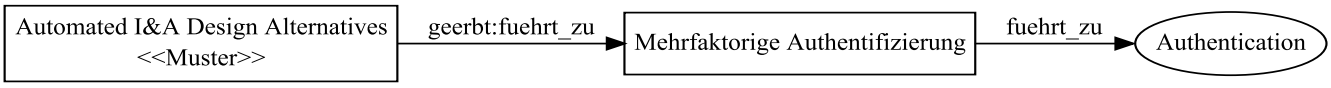
Gruppe(n)

- Anzahl der Authentifizierungskriterien

Kurzbeschreibung

Das Pattern "Mehrfaktorige Authentifizierung" beschreibt den Ansatz, mehrere Authentifizierungsverfahren gemeinsam zu nutzen, um die Qualität des Identitätsnachweises zu verbessern. Mehrfaktorig bedeutet hierbei, dass die Verfahren optimalerweise auf mehr als einem der Faktoren "Wissen", "Besitzen" usw. basieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Mehrfaktorige Authentifizierung -> Authentication** : komplexes Credential
  Mehrfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von mehr als einem Authentifizierungsverfahren. Die mindestens zwei verschiedenen Authentifizierungsverfahren sollen verschiedenartige Credentials aufweisen.
  Beispiel: Passwort (Wissen) + RSA-Token-ID (Besitz).

- **Automated I&A Design Alternatives -> Anzahl der Authentifizierungskriterien** :
  Anzahl der Authentifizierungskriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Je nach Sicherheitsbedarf der Applikation muss ein Subjekt ein oder mehrere Credentials richtig befüllen können um sich erfolgreich zu Authentisieren.

## *117. Memento*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 283-291

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 414-415

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Flexibilität

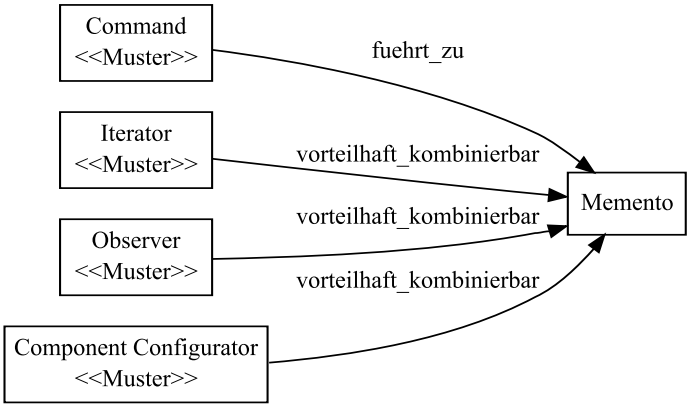
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Pattern "Memento" beschreibt einen Ansatz um Snapshots von Objektzuständen bereitzustellen.

Beziehungen - Grafik




Beziehungen - Detail

- **Command -> Memento** : Erreichung von Idempotenz
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships;
  Quelle: Design Patterns; Gamma et al.; S. 239; Punkt 3.
- **Iterator -> Memento** : Zur Speicherung des Iterationsstatus
  Quelle: Design Patterns; Gamma et al.; S. 271
  *"Memento is often used in conjunction with the Iterator pattern. An iterator can use a memento to capture the state of an iteratin. The iterator stores the memento internally."*
- **Observer -> Memento** :
  Quelle: POSA 4; Buschmann et al.; S. 414
- **Component Configurator -> Memento** :
  Quelle: POSA 4; Buschmann et al.; S. 414

## *118. Message Translator*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 85-94

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Flexibilität

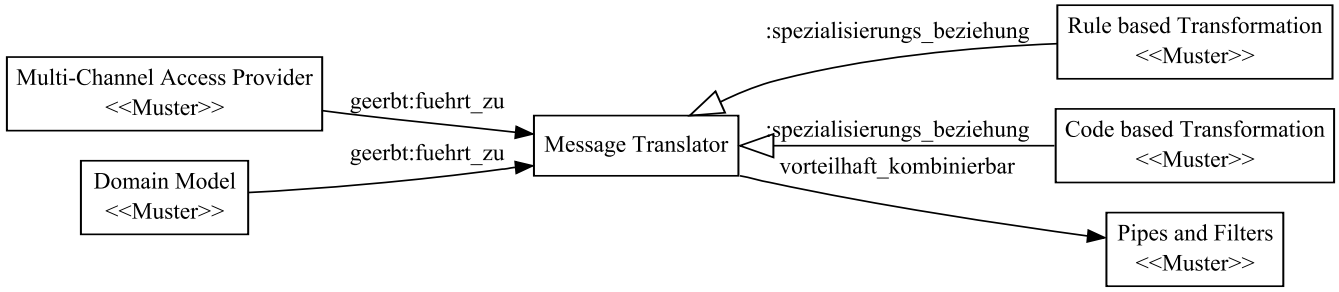
Gruppe(n)

- Datentransformation

Kurzbeschreibung

Das Pattern "Message Translator" beschreibt allgemein das Verfahren Nachrichten von einer Form (z. B. der im Sender-Prozess generierten Form) in eine andere Form (die von Empfänger erwartet wird) zu transformieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Message Translator -> Rule based Transformation** :
  Die Rule based Transformation ist eine spezielle Ausprägung des Message-Translator-Patterns.
- **Message Translator -> Code based Transformation** :
  Die Code based Transformation ist eine spezielle Ausprägung des Message-Translator-Patterns.
- **Message Translator -> Pipes and Filters** : Verketten von Message Translators
  Quelle: Enterprise Integration Patterns; Hohpe; S. 89
- **Multi-Channel Access Provider -> Datentransformation** : Umsetzung von Nachrichten in unterschiedliche Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zur Umsetzung des Multi-Channel-Access-Provider-Pattern wird die Verwendung von mindestens einem Pattern der Gruppe Datentransformation benötigt, um die verschiedenen Zugriffsarten durch jeweils geeignet umgesetzte Nachrichten zu unterstützen.
- **Domain Model -> Datentransformation** : Wenn: Rich Domain Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 117, Patterns of Enterprise Application Architecture, Folwer
  *“A rich Domain Model is better for more complex logic, but is harder to map to the database. A simple Domain Model can* [...]*, whereas a rich Domain Model requires Data Mapper.”*

## *119. Messaging*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 53

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

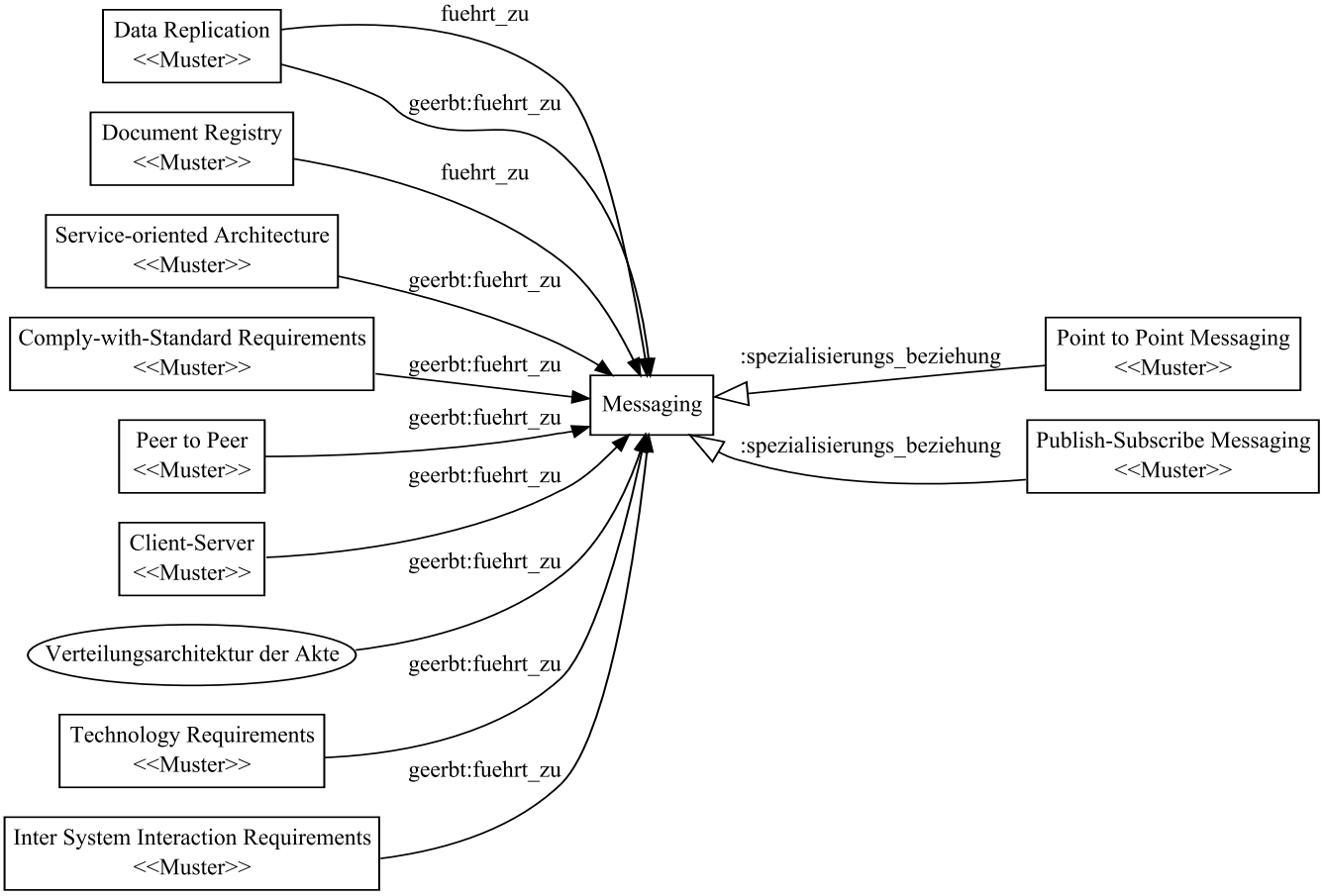
Gruppe(n)

- Asynchrone Kommunikation

Kurzbeschreibung

Das Pattern "Messaging" beschreibt in allgemeiner Form eine auf asynchronem Nachrichtenversand basierende Softwarearchitektur.

Beziehungen - Grafik



Beziehungen - Detail

- **Data Replication -> Messaging** : mögl. Umsetzung mit Messaging
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion;
  *"There are many different ways to implement data replication. For example,* [...] *or we can use message-oriented middleware to transport data records inside messages."*
- **Document Registry -> Messaging** : Asynchrones registrieren von Dokumenten
- **Messaging -> Point to Point Messaging** : Spezialform
- **Messaging -> Publish-Subscribe Messaging** : Spezialform
- **Data Replication -> Asynchrone Kommunikation** : verwendet
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion; „*There are many different ways to implement data replication. For example,* [...]*"* Alle der an dieser Stelle aufgeführten Methoden sind der asynchronen Kommunikation zuzuordnen.
- **Service-oriented Architecture -> Asynchrone Kommunikation** : für asynchrone Dienste
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: SOA in Practice, Josuttis; S. 125; Message Exchange Pattern: One-Way
  Quelle: SOA in Practice, Josuttis; S. 126; 10.2.3 Request/Response Versus Two One-Way Messages
  Quelle: SOA in Practice, Josuttis; S. 128; Message Exchange Pattern: Request/Callback
  Quelle: SOA in Practice, Josuttis; S. 129; Message Exchange Pattern: Publish/Subscribe
- **Comply-with-Standard Requirements -> Asynchrone Kommunikation** : wenn im Standard gefordert
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Peer to Peer -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Peer-to-Peer-Kommunikation ist mit den Mitteln asynchroner Kommunikation realisierbar.
- **Client-Server -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Client-Server-Systeme sind mit Mitteln asynchroner Kommunikation realisierbar
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.

- **Technology Requirements -> Asynchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Inter System Interaction Requirements -> Asynchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.

## *120. Meta-Directory*

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Anzahl der Authentifizierungsquellen
- Identitätsmanagement

Kontext

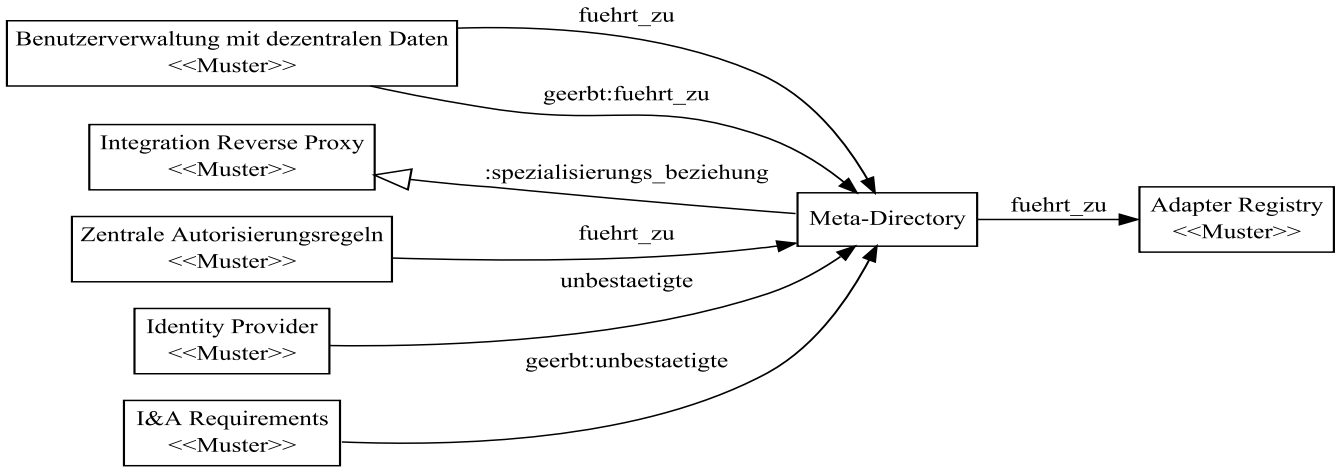
Benutzer werden in mehreren Datenquellen verwaltet, d. h. es existieren mehrere Benutzerverzeichnisse.

Problem
Viele Anwendungen können für die Authentifizierung von Benutzern zu einem Zeitpunkt nur für die Verwendung genau einer Benutzerdatenquelle konfiguriert werden.

Lösung
Es wird ein Meta-Benutzerverzeichnis geschaffen, das nach dem Prinzip eines Integration Reverse Proxy eine Menge von Verzeichnisdiensten zu einem virtuellen Benutzerverzeichnis integriert. Die Benutzer werden dadurch in ein virtuell gemeinsames Verzeichnis (Meta-Directory) zusammengefasst. Dieses Meta-Directory kann anschließend anstelle der einzelnen Verzeichnisse als Benutzerdatenquelle der Anwendungen verwendet werden.

Beispiel
OpenLDAP als MetaDirectory: siehe http://www.openldap.org/conf/odd-wien-2003/ando.pdf

Beziehungen - Grafik




Beziehungen - Detail

- **Benutzerverwaltung mit dezentralen Daten -> Meta-Directory** : Benötigt zur Erreichung einheitlicher Benutzer
  Eine Kombination der Benutzerverwaltung mit dezentral gespeicherten Daten mit dem Meta-Directory-Pattern ermöglicht eine Integration von Authentifizierungsdaten-quellen zu einem virtuell zentralen Authentifizierungsdienst.
- **Integration Reverse Proxy -> Meta-Directory** :
  Ein Meta-Directory ist ein Integration Reverse Proxy für den transparenten Zugriff auf mehrere Benutzerdatenquellen.
- **Zentrale Autorisierungsregeln -> Meta-Directory** : Benötigt
  Zur Umsetzung zentral verwalteter Autorisierungsregeln sind über alle Systemknoten hinweg eindeutige Benutzer notwendig um das zugrunde liegende Regelwerk eindeutig zu halten. Aus diesem Grund wird zur Umsetzung zentraler Autorisierungsregeln entweder das zentrale Benutzerverwaltung- oder das Meta-Directory-Pattern benötigt.
- **Meta-Directory -> Adapter Registry** : Umsetzbar durch
  Werden Benutzerdatenquellen mit unterschiedlichen Schnittstellen oder Standards verwendet, so kann es notwendig sein, für die Integration der Datenquellen unterschiedliche Adapter-Implementierungen zu verwenden. Der Integration Reverse Proxy kann dann unter Verwendung einer Adapter Registry implementiert werden.
- **Benutzerverwaltung mit dezentralen Daten -> Identitätsmanagement** : Zur Integration unterschiedlicher Benutzerstämme
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Pattern Benutzerverwaltung mit dezentralen Daten beschreibt einen Lösungsansatz der darauf basiert, dass jede Organisation ihre Benutzer selbst verwaltet und selbst in einem oder mehreren eigenen Systemen zur Benutzerverwaltung speichert. Das ist unter anderem der Fall, wenn verschiedene Primärsysteme mit jeweils eigener Benutzerverwaltung verwendet werden. Sollen derartige Systeme zu einem komplexen Systemverbund integriert werden, so ist die Verwendung von Patterns der Gruppe Identitätsmanagement zwangsläufig.

- **I&A Requirements -> Anzahl der Authentifizierungsquellen** : Beeinflusst die Auswahl
  Geerbte Beziehung: unbestaetigte_beziehung

## *121. Metadata-based Access Control*

Quellen

**Paper: A pattern system for access control, Priebe et al.**

A pattern system for access control; Torsten Priebe, Eduardo B. Fern, Jens I. Mehlau, Günther Pernul; in Research Directions in Data and Applications Security XVIII, C. Farkas and P. Samarati (Eds.), Procs of the 18th. Annual IFIP WG 11.3 Working Conference on Data and Applications Security
Online verfügbar unter:
http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.114.7637&rank=1

Schwerpunkt(e)

- Sicherheit

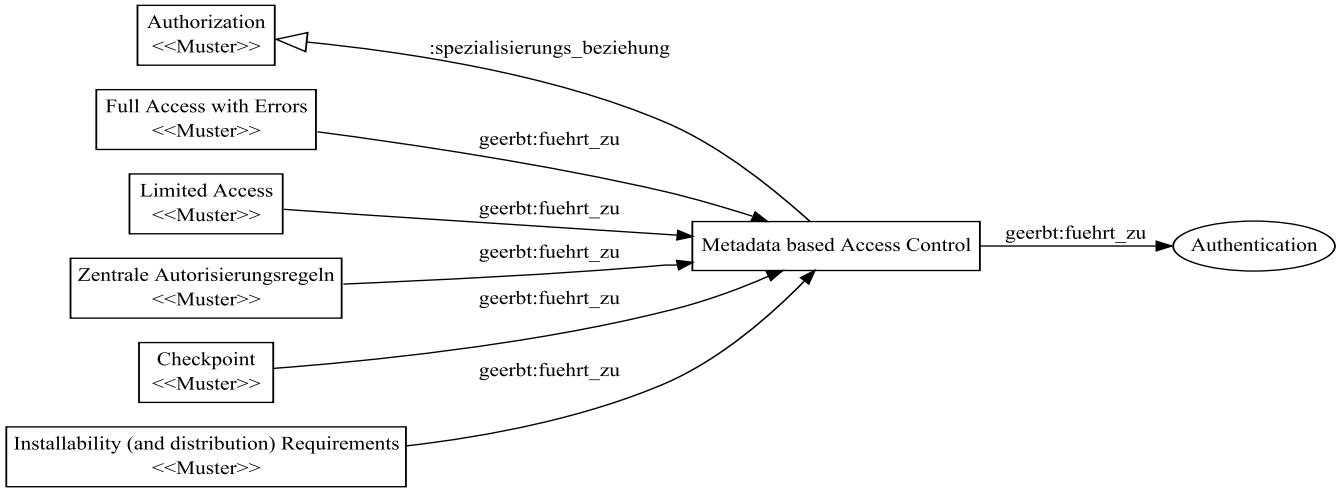
Gruppe(n)

- Access Control or Authorization

Kurzbeschreibung

Das Pattern "Metadata-based Access Control" beschreibt einen Ansatz der Regelung von Zugriffsrechten auf der Basis der Metadaten des zu schützenden Objekts.

Beziehungen - Grafik




Beziehungen - Detail

- **Full Access with Errors -> Access Control or Authorization** :
  Berechtigungsermittlung

Geerbte Beziehung: fuehrt_zu_beziehung
Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.

- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Geerbte Beziehung: fuehrt_zu_beziehung
  Aktenbezogene Autorisierungskonzepte lassen sich durch Kombination von Patterns der Gruppe Access Control or Authorization umsetzen.
  Bei der Verwendung zentraler Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.
- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al., S. 287;
  *"*[...]*, a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*
- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.

## *122. Microkernel*

Quellen

**POSA 1, A System of Patterns, Buschmann**

S. 171-192

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Flexibilität

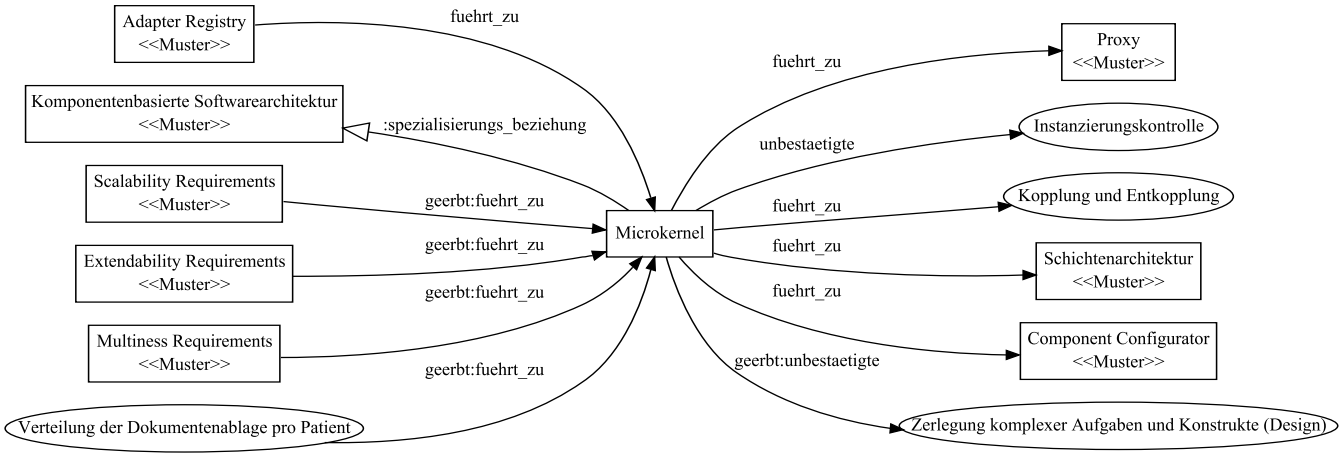
Gruppe(n)

- Grundlegende Architekturstile
- Flexibilitaetserhoehende Architekturmuster
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kurzbeschreibung

Das Microkernel-Pattern beschreibt das Konzept einer Softwarearchitektur, die aus einer kleinen Kern-Komponente besteht, die durch verschiedene Erweiterungs-Komponenten zu einem vollwertigen System erweitert werden kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Adapter Registry -> Microkernel** : Erweiterbarkeitsprinzip
  Es ist vorteilhaft die Registry-Schicht intern nach den Prinzipien einer Microkernel-Architektur erweiterbar zu gestalten. Die Integration der Adaptoren in die Registry-Schicht folgt diesem Prinzip bereits.
- **Komponentenbasierte Softwarearchitektur -> Microkernel** : Spezialform
  Die Microkernelarchitektur ist eine Architekturform, bei der ein kleiner Kernbestandteil durch hinzufügen von Komponenten zu einem komplexen und vielfältig funktionalen Gesamtsystem erweitert wird.
- **Microkernel -> Proxy** : Implementierung der Adaptoren
  Quelle: POSA 1; Buschmann et al.; Abhängigkeitsdiagramm auf der letzten Seite (ohne Nummer);
  *"Microkernel: implementing adapters"* auf dem Pfeil der Beziehung zwischen Broker, Microkernel und Proxy.
- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen

Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.
Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.
- **Microkernel -> Schichtenarchitektur** : zur Gliederung in Schichten
  Es ist häufig sinnvoll, das Microkernel-Pattern intern in Schichten von Erweiterungen zu gliedern. Dennoch entsteht daraus eine Sonderform der Schichtenarchitektur, die sich durch ihre erhöhte Flexibilität von vielen anderen Formen der Umsetzung von Schichtenarchitekturen unterscheidet.
  Quelle: S. 192 in Pattern oriented Software Architecture, Band 1.
  Übersetzt ins Deutsche und sinngemäß wiedergegeben: *„Die Beziehung zwischen dem Schichtenarchitektur- und Microkernel-Pattern ist zwiespältig. Einerseits kann das Mikrokernel-Pattern als Variante des Schichtenarchitektur-Patterns verstanden werden* [...] *Andererseits können beide Patterns in manchen Anwendungsdomänen auch alternativ verwendet werden".*
- **Microkernel -> Component Configurator** : Komponentenkonfiguration zur Laufzeit
  Quelle: S. 490; Pattern oriented Software Architecture, Volume 4
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.
- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen

Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.

- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *123. Minimize Human Intervention*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 57-59

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

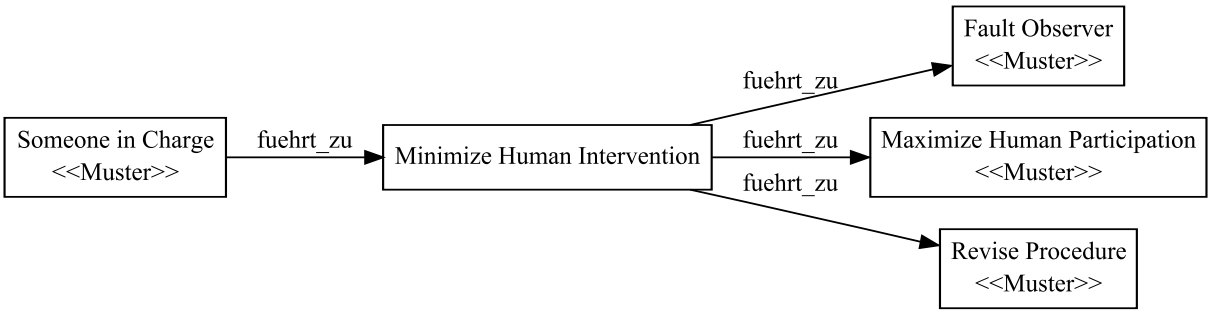
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Menschen machen Fehler und sind langsam. Aus diesen Gründen beschreibt das Pattern "Minimize Human Intervention" eine Empfehlung, die Notwendigkeit des Eingreifens durch menschliche Experten überall dort zu minimieren, wo eine zuverlässige automatische Behandlung von Fehlern möglich ist.

Beziehungen - Grafik




Beziehungen - Detail

- **Someone in Charge -> Minimize Human Intervention** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Minimize Human Intervention -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Minimize Human Intervention -> Maximize Human Participation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Minimize Human Intervention -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *124. Model View Controller*

Quellen

**Patterns of Enterprise Application Architecture, Fowler**

S. 330 ff.

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

**POSA 1, A System of Patterns, Buschmann**

S. 125 ff.

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Flexibilität

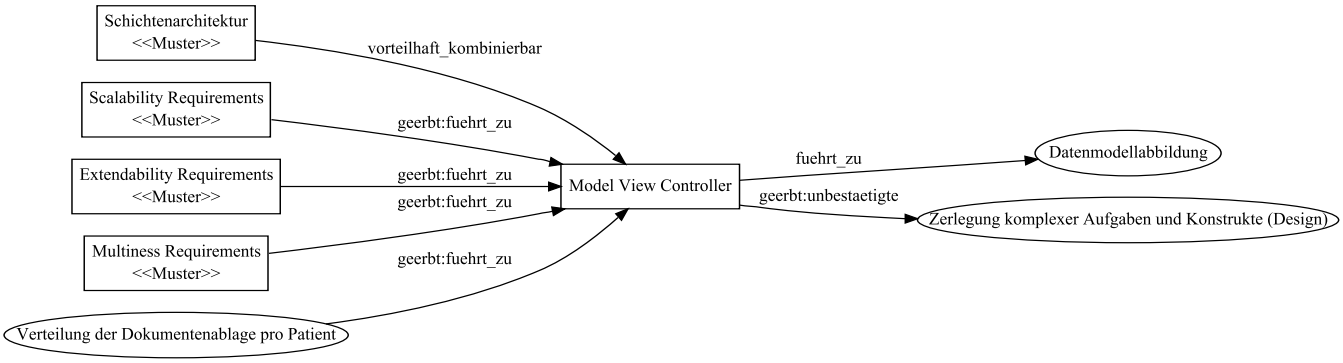
Gruppe(n)

- Grundlegende Architekturstile
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

## Kurzbeschreibung

Das Pattern "Model View Controller" beschreibt die Zerlegung des Codes von Benutzeroberflächen in die Bestandteile Model, View und Controller und führt so zu einer Trennung der Aspekte Darstellung, Verarbeitung und Abbildung des Datenmodells.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Schichtenarchitektur -> Model View Controller** : GUI-Schicht
  Ist ein System in Form einer Schichtenarchitektur aufgebaut, so kann die Schicht seiner Benutzeroberfläche intern nach dem Model-View-Controller-Pattern strukturiert sein. Die Existenz einer Schichtenarchitektur ist allerdings keine notwendige Voraussetzung für die Verwendung von Model View Controller.
- **Model View Controller -> Datenmodell-Abbildung** : Gestaltung des Bestandteils Model
  Der Bestandteil "Model" kann in verschiedenen Formen abgebildet werden. Die gängigste Form dabei ist das Domain Model. Das Domain-Model-Pattern ist Bestandteil der Gruppe Datenmodell-Abbildung. Auch die anderen Elemente der Gruppe können zur Umsetzung eines Model im Rahmen der Anwendung des Model-View-Controller-Pattern verwendet werden.
  Quelle: S. 330; Patterns of Enterprise Application Architecture; Fowler;
  *“In its most pure OO form the model is an object within a Domain Model. You might also think of* [...] *“*
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der

Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.

- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.
- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *125. Multi-Channel Access Provider*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Flexibilitaetserhoehende Architekturmuster

Kontext

Im Umfeld verteilter Krankenakten existiert eine Vielzahl teilweise konkurrierender Standards. Um flächendeckend interoperabel zu sein müssen die Knoten einer verteilten Krankenakte technisch geeignet sein, mit einer Vielzahl anderer Knoten zu kommunizieren. Dabei kann nicht ausgeschlossen werden, dass verschiedene Knoten verschiedene Standards unterstützen.

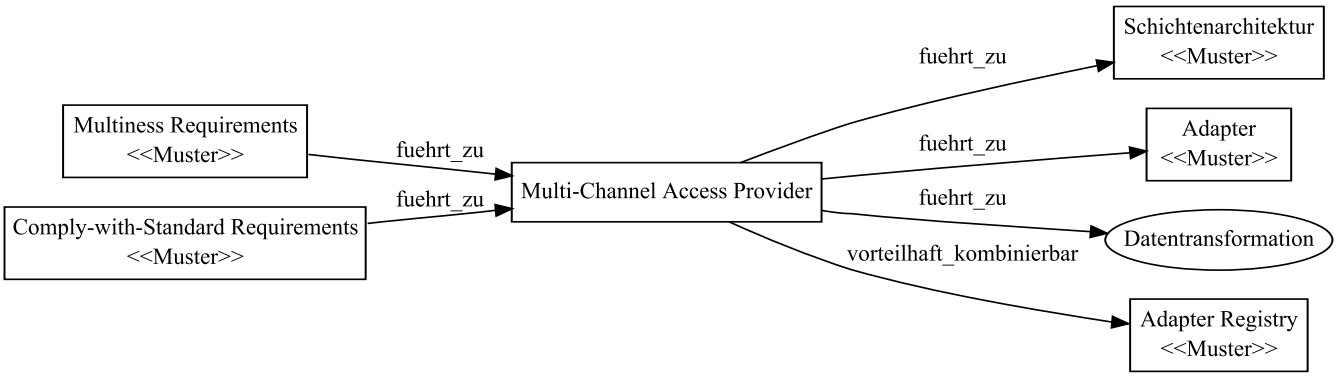
### Problem

Verschiedene Standards setzen im Allgemeinen auf unterschiedliche Kommunikationsprotokolle, unterschiedliche Nachrichtenformate und ggf. auch unterschiedliche zusätzliche Inhaltsbestandteile in den Nachrichten. Das bedeutet, dass auch wenn die der Kommunikation zugrunde liegende Kernfunktionalität die gleiche ist, für jeden zusätzlich unterstützten Standard zusätzliche Anforderungen an die jeweilige Schnittstelle hinzukommen.

### Lösung

Der Lösungsansatz Multi-Channel Access Provider besteht aus zwei Schichten (Schichtenarchitektur Muster), von denen in der unteren Schicht (Basisschicht) generisch die eigentliche Funktionalität implementiert wird, während die darüberliegende Schicht (Schnittstellenschicht) aus einer Vielzahl von Fassade/Adapter-Implementierungen besteht, die unter Verwendung der Methoden der Basisschicht die Verhaltensweisen der verschiedenen adressierten Standards adaptieren.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Multiness Requirements -> Multi-Channel Access Provider** : Hohe Anforderungen umsetzbar durch
  Hohe Anforderungen an die Schnittstellenvielfalt können durch die Anwendung des Multi-Channel-Access-Provider-Patterns umgesetzt werden.
- **Comply-with-Standard Requirements -> Multi-Channel Access Provider** : wenn synchrone und asynchrone Standards in Anforderungen
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen. Wenn eine Vielzahl von Standards gefordert wird, dann eignet sich ggf. die Anwendung eines Musters zur Bewältigung dieser Vielfältigkeit.
- **Multi-Channel Access Provider -> Schichtenarchitektur** : Als Bestandteil
  Das Multi-Channel-Access-Provider-Pattern ist selbst in Form einer Schichtenarchitektur aufgebaut. Es besteht aus den Schichten Basisschicht und Schnittstellenschicht.
- **Multi-Channel Access Provider -> Adapter** : Adapter ist Bestandteil der Schnittstellenschicht
  Das Multi-Channel-Access-Provider-Pattern beinhaltet als Bestandteile seiner

Schnittstellenschicht Adaptoren, die entsprechend dem Adapter-Pattern implementierbar sind.

- **Multi-Channel Access Provider -> Datentransformation** : Umsetzung von Nachrichten in unterschiedliche Standards
  Zur Umsetzung des Multi-Channel-Access-Provider-Pattern wird die Verwendung von mindestens einem Pattern der Gruppe Datentransformation benötigt, um die verschiedenen Zugriffsarten durch jeweils geeignet umgesetzte Nachrichten zu unterstützen.
- **Multi-Channel Access Provider -> Adapter Registry** : als Datenquelle AdapterRegistry als Datenquelle für Multi-Channel Access Provider; Access Provider als Mittel die geschaffene vereinheitlichte Schnittstelle in einer Vielzahl von Standards zur Verfügung zu stellen.

## *126. Multi-Lingual Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 272-274

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Flexibilität

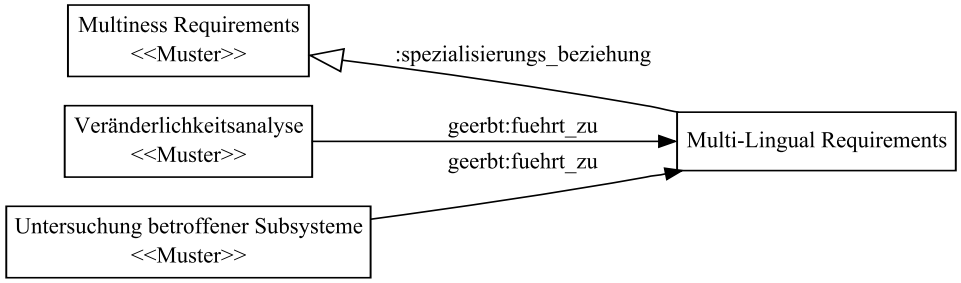
Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

Kurzbeschreibung

Das Pattern "Multi-Lingual Requirements" beschreibt die Definition von Anforderungen an die Mehrsprachigkeit von Softwaresystemen.

Beziehungen - Grafik



Beziehungen - Detail

- **Multiness Requirements -> Multi-Lingual Requirements** :
  Quelle: Software Requirement Patterns; Withall; S. 272; Related Patterns
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung.
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *127. Multiness Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 261-272

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Flexibilität

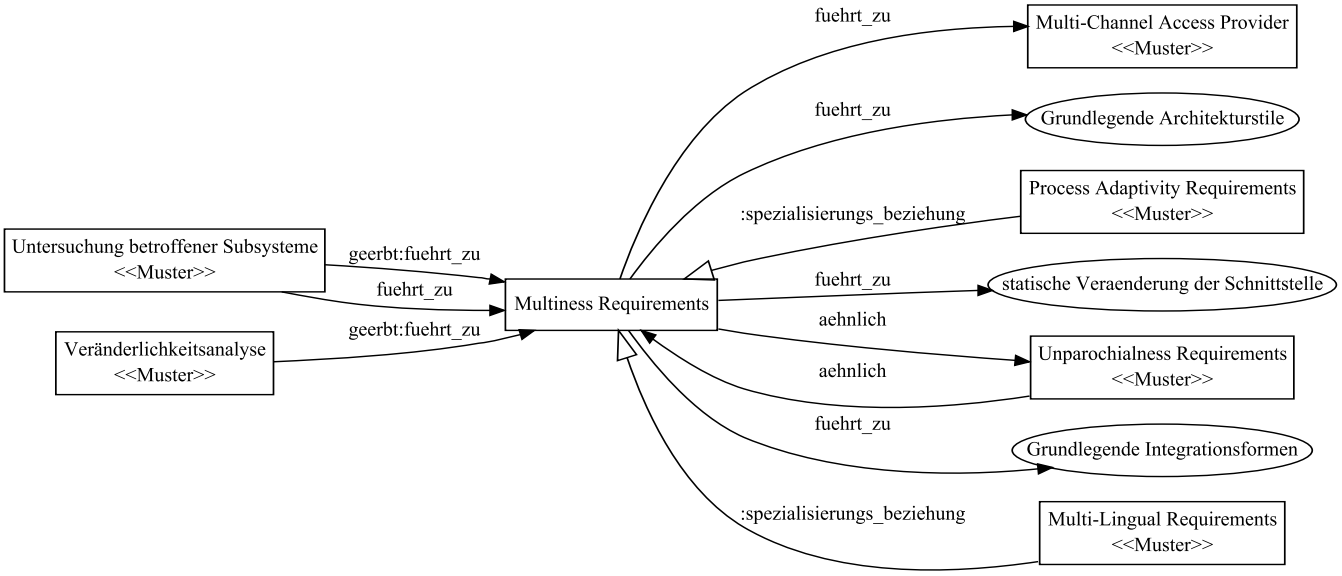
Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

Kurzbeschreibung

Das Pattern "Multiness Requirements" wird verwendet, wenn ein System gleichzeitig mehrere verschiedene Dinge (Standards, Datenbanken, Quellsysteme, Benutzeroberflächentechnologien usw.) unterstützen muss. Es unterstützt dabei, diese Eigenschaften in Form von Anforderungen festzuhalten.

Beziehungen - Grafik




Beziehungen - Detail

- **Untersuchung betroffener Subsysteme -> Multiness Requirements** : beeinflusst direkt
  Das Ergebnis der Untersuchung betroffener Subsysteme beeinflusst direkt die Anforderungen an die Vielfältigkeit der Schnittstellen.
- **Unparochialness Requirements -> Multiness Requirements** : entweder oder
  Quelle: Software Requirement Patterns; Withall; S. 261: *"Do not use the multiness requirement pattern when the system need support only one from a range of alternatives; use the unparochialness requirement pattern in that case"*
- **Multiness Requirements -> Multi-Channel Access Provider** : Hohe Anforderungen umsetzbar durch
  Hohe Anforderungen an die Schnittstellenvielfalt können durch die Anwendung des Multi-Channel-Access-Provider-Patterns umgesetzt werden.
- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.
- **Multiness Requirements -> Process Adaptivity Requirements** : Vielfältigkeit der Prozesse
  Die an Prozesse gestellten Vielfältigkeitanfoderungen werden durch das Process-adaptivitiy-Requirements-Pattern beschrieben.
- **Multiness Requirements -> statische Veraenderung der Schnittstelle** : Indirekt: Abbildung auf Ziel-Schnittstellen
  Grundsätzlich werden Muster zur statischen Veränderung der Schnittstelle verwendet

um einzelne der Multiness Requirements zu erfüllen. Es ist aber zusätzlich ein größeres Pattern (Ebene Architektur) wie z. B. das Multi-Channel-Access-Provider-Pattern notwendig um die Multiness Requirements in einer konsistenten Architektur abzubilden.

- **Multiness Requirements -> Unparochialness Requirements** : entweder oder
  Quelle: Software Requirement Patterns; Withall; S. 261: *"Do not use the multiness requirement pattern when the system need support only one from a range of alternatives; use the unparochialness requirement pattern in that case"*
- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.
- **Multiness Requirements -> Multi-Lingual Requirements** :
  Quelle: Software Requirement Patterns; Withall; S. 272; Related Patterns
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *128. Non-Repudiation Requirements*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 396-402

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

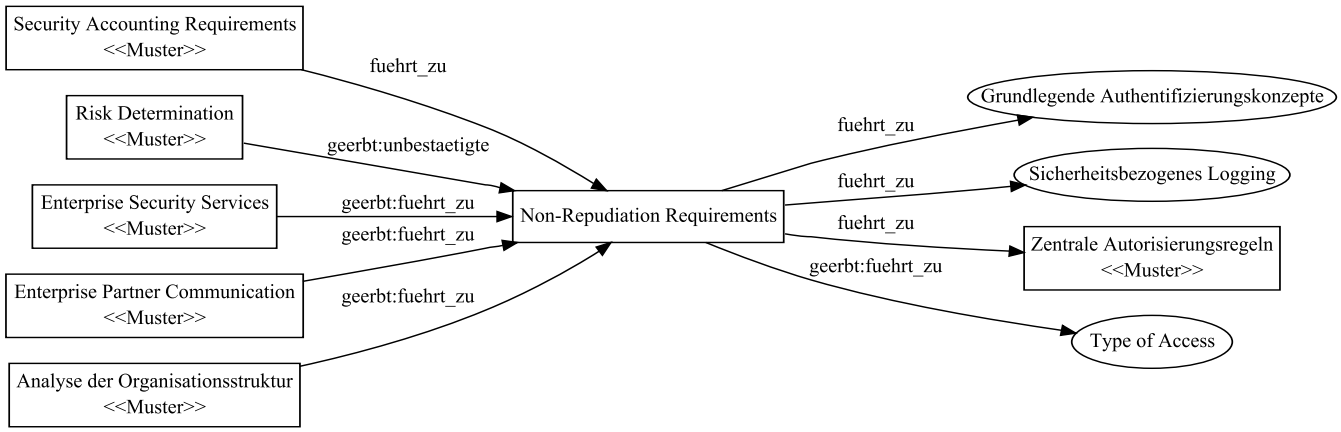
### Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

### Kurzbeschreibung

Das Pattern "Non-Repudiation Requirements" dient der Definition von Anforderungen bezüglich der Nachweisbarkeit und Nichtabstreitbarkeit von Protokoll-Inhalten zu sicherheitsrelevanten Vorfällen.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Security Accounting Requirements -> Non-Repudiation Requirements** : Wenn verschärfte Anforderungen an Beweissicherheit vorliegen
  Quelle: Security Patterns, Schumacher et al.; S. 396 u. 360 (kombiniert betrachtet)
- **Non-Repudiation Requirements -> Grundlegende Authentifizierungskonzepte** : Authentifizierungsanforderungen für Nichtabstreitbarkeit
  Quelle: Security Patterns; Schumacher et al.; S. 397; Abbildung "Common model for non-repudiation".
- **Non-Repudiation Requirements -> Sicherheitsbezogenes Logging** : Erweiterte Anforderungen für das Logging
  Wenn Erkenntnisse über Sicherheitsvorfälle nicht abstreitbar sein sollen, muss deren Unverfälschbarkeit oder Unverfälschtheit nachweisbar sein. Aus diesem Grund erweitern mögliche "Non-Repudiation Requirements" die Anforderungen für das Logging.
- **Non-Repudiation Requirements -> Zentrale Autorisierungsregeln** : Dokumentation früherer Berechtigungen
  Durch zentral verwaltete Autorisierungsregeln wird die Historisierung der Berechtigungen von Benutzer möglich bzw. erleichtert.
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten

gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.

- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** :
  Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *129. Object Relational Mapping*

Quellen

**Patterns of Enterprise Application Architecture, Fowler**

Ausprägung von Object Relational Mapping
Name: Data Mapper
S. 165-181

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

Schwerpunkt(e)

- Flexibilität

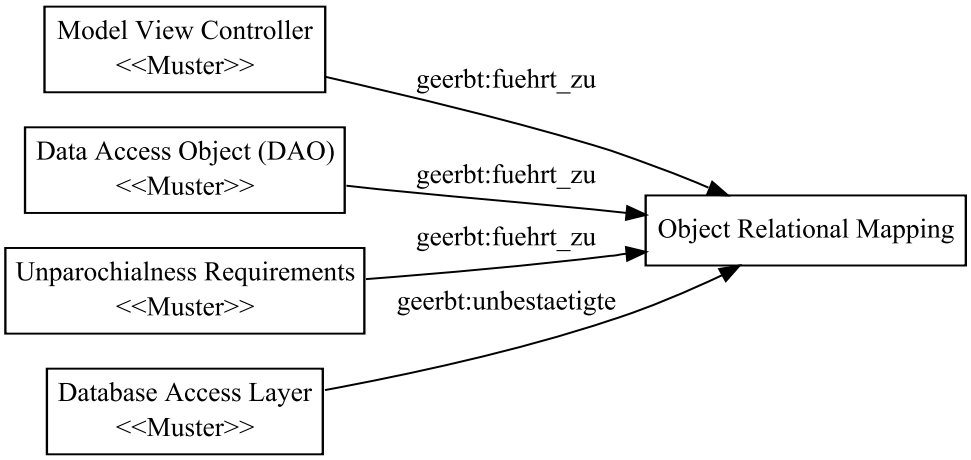
Gruppe(n)

- Datenmodell-Abbildung

Kurzbeschreibung

Das Pattern "Object Relational Mapping" beschreibt ein Vorgehen zur Abbildung von Datensätzen relationaler Datenbanken auf Objekte eines objektorientierten Modells.

Beziehungen - Grafik




Beziehungen - Detail

- **Model View Controller -> Datenmodell-Abbildung** : Gestaltung des Bestandteils Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Der Bestandteil "Model" kann in verschiedenen Formen abgebildet werden. Die gängigste Form dabei ist das Domain Model. Das Domain-Model-Pattern ist Bestandteil der Gruppe Datenmodell-Abbildung. Auch die anderen Elemente der Gruppe können zur Umsetzung eines Model im Rahmen der Anwendung des Model-View-Controller-Pattern verwendet werden.

Quelle: S. 330; Patterns of Enterprise Application Architecture; Fowler;
*"In its most pure OO form the model is an object within a Domain Model. You might also think of* [...]*"*

- **Data Access Object (DAO) -> Datenmodell-Abbildung** : Verwendet eine Ausprägung von
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das DAO-Pattern dient der Vereinfachung des Zugriffs auf Datenbestände. Diese Datenbestände selbst können entsprechend der Prinzipien eines der Patterns der Gruppe Datenmodell-Abbildung aufgebaut sein.
- **Unparochialness Requirements -> Datenmodell-Abbildung** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Unparochialness-Requirements-Pattern dient der Definition von Anforderungen bezüglich der Fähigkeit eines Systems, sich veränderlichen Umgebungsbedingungen anzupassen. Die verschiedenen Patterns der Gruppe Datenmodell-Abbildung führen zu unterschiedlich flexiblen Datenmodellen. Während z. B. das Entity-Attribute-Value-Pattern zu einem leicht veränderbaren Datenmodell führt, wird beim Domain-Model-Pattern das Datenmodell relativ statisch, dafür aber einfacher verwendbar, abgelegt.
- **Database Access Layer -> Datenmodell-Abbildung** : Abhängig
  Geerbte Beziehung: unbestaetigte_beziehung
  Die konkrete Gestaltung der Database Access Layer ist abhängig von der gewählten Form der Datenmodell-Abbildung.

## *130. Observer*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 293-303

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

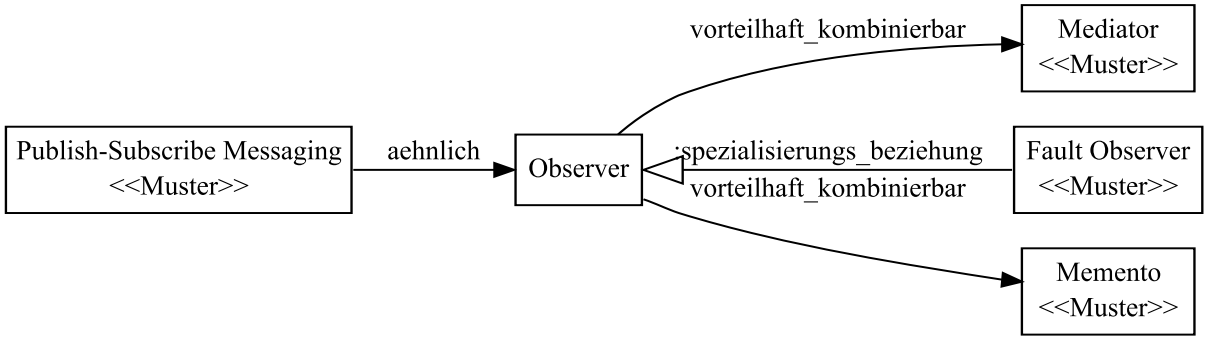
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Observer-Pattern beschreibt einen Mechanismus, der es einem Objekt erlaubt, sich für den Empfang von Ereignissen eines anderen Objekts zu registrieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Publish-Subscribe Messaging -> Observer** :
  Das Observer-Pattern und das Publisher-Subscriber-Pattern, das speziell den Anwendungsfall Messaging des Observer-Patterns beschreibt, sind bezogen auf ihre Auswirkung ähnlich.
- **Observer -> Mediator** : Behandlung komplexer Abhängigkeiten zwischen Colleagues
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Quelle: Design Patterns; Gamma et al.; S. 282
  *„Colleagues can communicate with the mediator using the Observer pattern.“*
- **Observer -> Fault Observer** : Spezialisierung: Fehlerüberwachung
  Anwendung der Prinzipien des Observer-Patterns auf die Überwachung von Fehlern.
- **Observer -> Memento** :
  Quelle: POSA 4; Buschmann et al.; S. 414

## *131. Overload Toolboxes*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 184-186

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

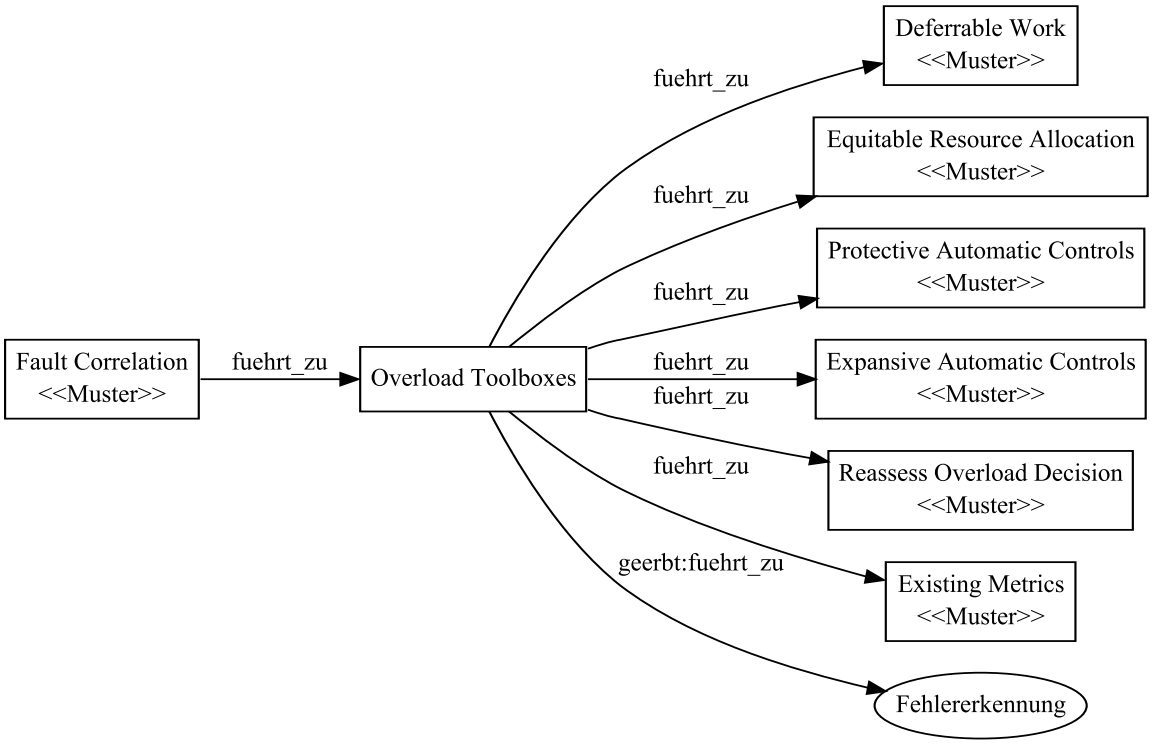
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Overload Toolboxes" beschreibt die Empfehlung, verschiedene Werkzeuge oder Strategien für die verschiedenen denkbaren Überlast-Szenarios vorzubereiten.

Beziehungen - Grafik




Beziehungen - Detail

- **Fault Correlation -> Overload Toolboxes** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Overload Toolboxes -> Deferrable Work** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Overload Toolboxes -> Equitable Resource Allocation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Overload Toolboxes -> Protective Automatic Controls** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Overload Toolboxes -> Expansive Automatic Controls** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Overload Toolboxes -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Overload Toolboxes -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *132. Packet Filter Firewall*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 405-410

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

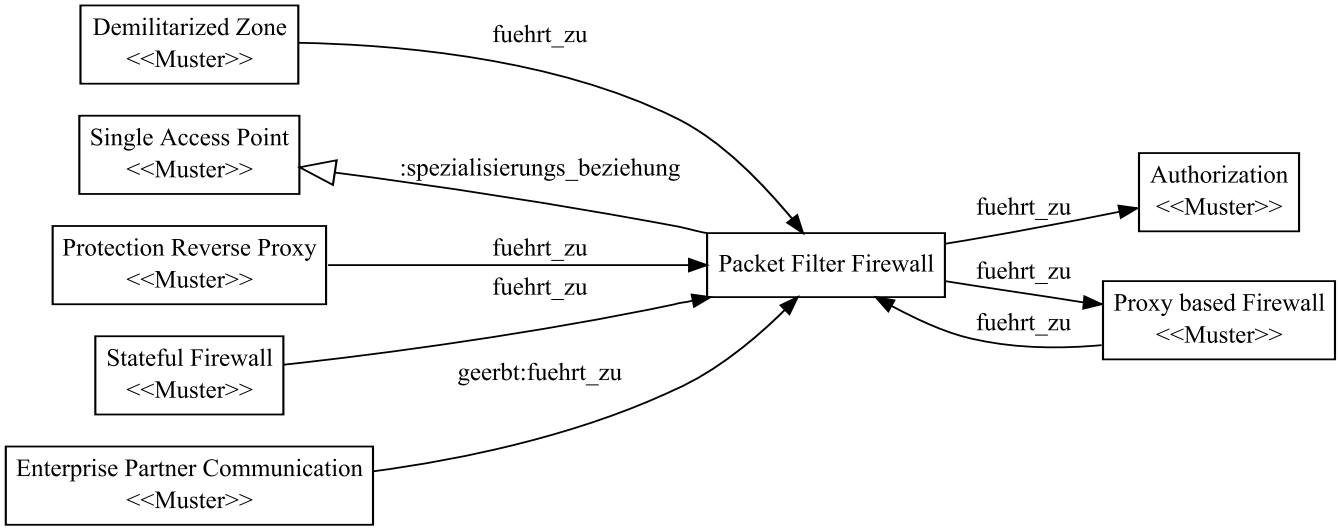
Gruppe(n)

- Access Restriction

Kurzbeschreibung

Das Pattern "Packet Filter Firewall" beschreibt den Ansatz, unberechtigte Netzwerkzugriffe durch das regelbasierte Ausfiltern von Paketen des IP-Protokolls zu unterbinden.

Beziehungen - Grafik




Beziehungen - Detail

- **Demilitarized Zone -> Packet Filter Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Single Access Point -> Packet Filter Firewall** :
  Quelle: Security Patterns; Schumacher et al.; S. 410;
  *"This pattern is also a special case of Single Access Point"*

- **Protection Reverse Proxy -> Packet Filter Firewall** :
  Quelle: Security Patterns; Schumacher et al.; S. 458;
  ***Solution***
  *"*[...] *Two Packet Filter Firewalls ensure that no external network traffic reaches the real Web server."*
- **Proxy based Firewall -> Packet Filter Firewall** : Filterung auf Netzwerkebene
  Quelle: Security Patterns; Schumacher et al.; S.416
  *"This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall."*
- **Stateful Firewall -> Packet Filter Firewall** : Üblicherweise kombiniert
  Quelle: Security Patterns; Schumacher et al; S. 421
  *"This firewall* [Stateful Firewall] *is usually combined with one or both of the previous types of firewalls, Packet Filter Firewall and Proxy-Based Firewall."*
- **Packet Filter Firewall -> Authorization** :
  Regelbasierte Authorisierung von Paketen.
  Quelle: Security Patterns; Schumacher et al.; S. 410
- **Packet Filter Firewall -> Proxy based Firewall** : Wenn Firewall auf Applikationsebene benötigt
  Wenn die Verwendung einer Packet Filter Firewall nicht ausreichend ist, weil die Bedrohung auf Angriffen basiert, die auf einer höheren Protokollebene stattfinden, kann ggf. mit einer Proxy based Firewall abhilfe geschaffen werden.
  Quelle: Security Patterns; Schumacher et al.; S. 411
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *133. Password Design and Use*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 217-228

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

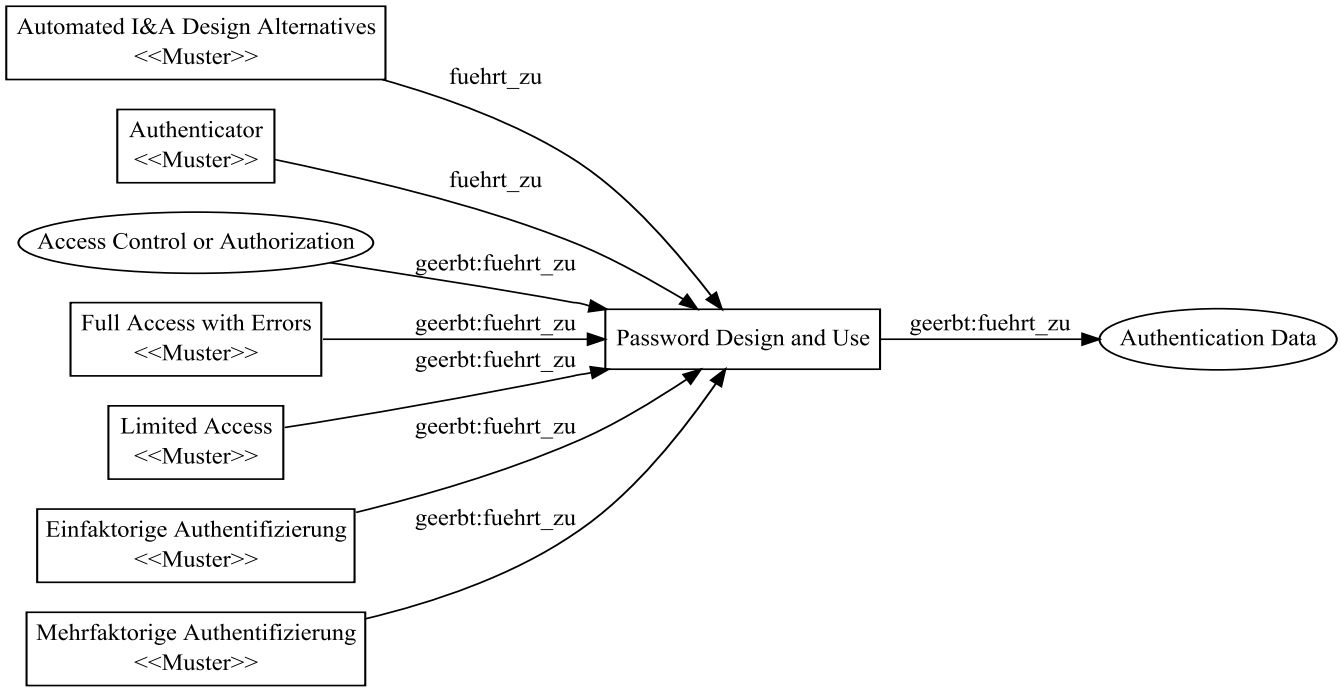
Gruppe(n)

- Authentication

## Kurzbeschreibung

Das Pattern "Password Design and Use" beschreibt Empfehlungen für die Gestaltung und den Einsatz von Passwörtern.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Automated I&A Design Alternatives -> Password Design and Use** : I&A Methode = Passwort
  Quelle: Security Patterns; Schumacher et al.; S. 209; 2. Satz im Abschnitt Solution.
- **Authenticator -> Password Design and Use** : wenn Benutzername/Passwort verwendet wird
  Wenn Passwörter zur Authentifizierung verwendet werden, so führt die Anwendung des Authenticator-Patterns zur Verwendung des Password-Design-and-Use-Patterns.
  Quelle der Beziehung: S. 327, Security Patterns, Schumacher et al.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.
- **Full Access with Errors -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Full Access with Errors beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.

Quelle: Security Patterns, Schumacher et al.; S: 309
Beschreibt die Beziehung zu Identifcation und Authentication Patterns.

- **Limited Access -> Authentication** : Ermittlung des Benutzers
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen durch Limited Access beschränkten Zugriff durchzusetzen, ist eine verlässliche Identifikation des nutzenden Subjekts (Benutzer oder Maschine) notwendig. Diese Identifikation erfolgt durch die Anwendung eines Authentifizierungsverfahrens.
- **Einfaktorige Authentifizierung -> Authentication** : Authentifizierung mit einfachem Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Einfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von nur einem Authentifizierungsverfahren. Typische Beispiele sind:
  - Zugang zum System durch Benutzername/Passwort
  - Zugangssicherung durch Fingerabdruck-Scan
- **Mehrfaktorige Authentifizierung -> Authentication** : komplexes Credential
  Geerbte Beziehung: fuehrt_zu_beziehung
  Mehrfaktorige Authentifizierung bezeichnet die gleichzeitige Verwendung von mehr als einem Authentifizierungsverfahren. Die mindestens zwei verschiedenen Authentifizierungsverfahren sollen verschiedenartige Credentials aufweisen. Beispiel: Passwort (Wissen) + RSA-Token-ID (Besitz).
- **Authentication -> Authentication Data** : als Datenquelle
  Geerbte Beziehung: fuehrt_zu_beziehung
  Für die Authentifizierung von Benutzern oder Geräten werden Authentifizierungsdaten z. B. in Form von Credentials benötigt.

## *134. Patient based Access Control*

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Krankenaktenspezifische Berechtigungskonzepte

Kontext

Verschiedene medizinische Einrichtungen oder Organisationseinheiten einer Einrichtung sollen eine gemeinsame Behandlung von Patienten durchführen. Die Vorbedingung, dass Patienten und medizinisches Personal über Einrichtungsgrenzen hinweg im System eindeutig identifiziert werden könne, ist erfüllt.

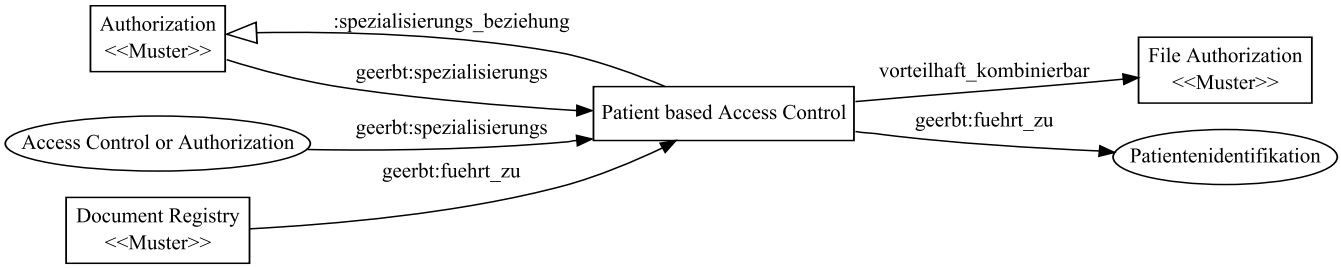
Problem
Nur fachlich berechtigtes Personal darf auf Dokumente zu einem Patienten zugreifen können.

Lösung

Der Patient erteilt mit seiner Aufnahme den an der Behandlung beteiligten Personen die Erlaubnis, alle Dokumente zu seiner Person einzusehen. Dadurch kann die Zugriffsberechtigung vereinfachend an den Patienten und nicht an einzelne Dokumente (File Authorization) oder Behandlungsfälle (Case based Access Control) gebunden werden.

Diese Vorgehensweise vereinfacht die Verwaltung der Berechtigungen und ist vor allem dann, wenn Datenschutzvorschriften nicht besonders detailliert oder nicht in vollem Umfang relevant sind, als Vorteilhafte, da wenig bürokratische Lösung zu betrachten.

Beziehungen - Grafik




Beziehungen - Detail

- **Authorization -> Patient based Access Control** : Granularität: Einzelner Patient
  Diese Beziehung ist abgeleitet von der belegten Beziehung zwischen Authorization und Role based Access Control.
- **Patient based Access Control -> File Authorization** : Mehrstufige Berechtigungen
  Patient based Access Control kann auch kombiniert mit File Authorization eingesetzt werden. So ist beispielsweise denkbar, den Zugriff auf Auflistungen vorhandener Dokumente durch Patient based Access Control zu realisieren, wärend das lesen oder ergänzen eines Dokuments durch File Authorization restriktiert wird.
- **Access Control or Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Krankenaktenspezifische Berechtigungskonzepte sind spezielle Autorisierungsmechanismen. Sie leiten sich von allgemeinen Autorisierungsmechanismen ab.
- **Authorization -> Krankenaktenspezifische Berechtigungskonzepte** :
  Geerbte Beziehung: spezialisierungs_beziehung
  Die Gruppe Krankenaktenspezifische Berechtigungskonzepte beinhaltet eine Menge von Patterns die spezifisch für den Einsatz in verteilten Krankenakten und eine Spezialisierung des Authorization-Patterns sind. (z. B. Case based Authorization)
- **Document Registry -> Krankenaktenspezifische Berechtigungskonzepte** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Kontext medizinischer Informationssysteme ist häufig bereits das Wissen über die Existenz eines Dokuments zu einem bestimmten Typ von Behandlung datenschutzrechtlich relevant. Aus diesem Grund ist es notwendig auch in einem

Verzeichnisdienst, der die Suche über verteilt abgelegte Dokumente ermöglicht, ein umfassendes Berechtigungssystem zu integrieren.

- **Krankenaktenspezifische Berechtigungskonzepte -> Patientenidentifikation** : Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im medizinischen Kontext ist die zentrale Entität i.d.R. der Patient. Seine eindeutige Identifizierung, auch über Systemgrenzen hinweg, ist die Basis der meisten medizinspezifischen Autorisierungsmechanismen.

## *135. Peer to Peer*

Quellen

**Peer: an Architectural Pattern, Amoretti et al.**

Peer: an Architectural Pattern; M. Amoretti, M. Reggiani, F. Zanichelli, G. Conte; In PLoP '05: Proceedings of the 2005 conference on Pattern languages of programs (2005) Online verfügbar unter:
http://hillside.net/plop/2005/proceedings/PLoP2005_mamoretti0_1.pdf
oder: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.374&rep=rep1&type=pdf

Schwerpunkt(e)

- Kommunikation

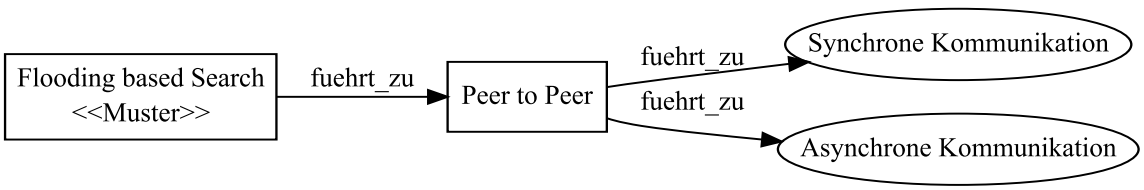
Gruppe(n)

- Rolle der Kommunikanten

Kurzbeschreibung

Das Pattern "Peer to Peer" beschreibt eine Architektur gleichberechtigter Knoten, die miteinander Kommunizieren und so ein verteiltes System zur Erfüllung einer gegebenen Aufgabe bilden.

Beziehungen - Grafik




Beziehungen - Detail

- **Flooding based Search -> Peer to Peer** : Benötigt zwangsläufig
  Flooding ist eine ausschließlich im Peer-to-Peer-Bereich anwendbare Art der Suche.
- **Peer to Peer -> Synchrone Kommunikation** : realisierbar mit
  Peer-to-Peer-Kommunikation ist mit mitteln synchroner Kommunikation realisierbar.

- **Peer to Peer -> Asynchrone Kommunikation** : realisierbar mit Peer-to-Peer-Kommunikation ist mit den Mitteln asynchroner Kommunikation realisierbar.

## *136. Persistent State Pattern*

Quellen

**PLoP2010**

Persistent State Pattern; A. Saude; R. Victorio; G. Coutinho; In PLoP '10: Proceedings of the 2010 conference on Pattern languages of programs (2010);
http://www.hillside.net/plop/2010/papers/saude.pdf

http://www.hillside.net/plop/2010

Schwerpunkt(e)

- Flexibilität

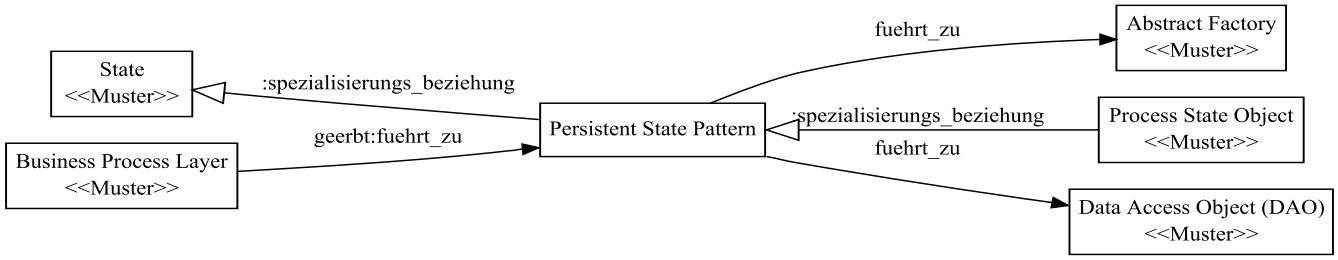
Gruppe(n)

- Verwaltung des Prozesszustands

Kurzbeschreibung

Das Persistent-State-Pattern behandelt die Verwaltung von für den Prozesszustand relevanten Informationen in persistenten Objekten.

Beziehungen - Grafik




Beziehungen - Detail

- **State -> Persistent State Pattern** :
  vgl. http://www.hillside.net/plop/2010/papers/saude.pdf
- **Persistent State Pattern -> Abstract Factory** : create and control a transaction lifecycle
  Das Abstract-Factory-Pattern ist als Transaction-Factory ein Bestandteil des Persistent-State-Pattern.
  Quelle: http://www.hillside.net/plop/2010/papers/saude.pdf Nr. 4.5 Structure und Nr. 4.7 Related Patterns
- **Persistent State Pattern -> Process State Object** :
  Das Process State Object ist eine Form des Persistent-State-Pattern.

- **Persistent State Pattern -> Data Access Object (DAO)** : Bestandteil
  Quelle: Quelle: http://www.hillside.net/plop/2010/papers/saude.pdf Nr. 4.5 Structure; *"Data Access Object (DAO): The DAO is an abstraction layer for the communication with the database"*
- **Business Process Layer -> Verwaltung des Prozesszustands** : notwendig zur Abbildung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen Geschäftsprozess durch Software abbilden zu können, muss das zugehörige IT-System Kenntniss über den Zustand einer jeden Prozess-Instanz besitzen. Auf Basis des Zustands wird vom System der nächste notwendige Bearbeitungsschritt ermittelt. Wie dieser Zustand technisch verwaltet wird kann unterschiedlich gelöst werden. Die Gruppe Verwaltung des Prozesszustands stellt mehrere Kandidaten zur Auswahl.

## *137. Pipes and Filters*

Quellen

**POSA 1, A System of Patterns, Buschmann**

S. 53-70

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Flexibilität

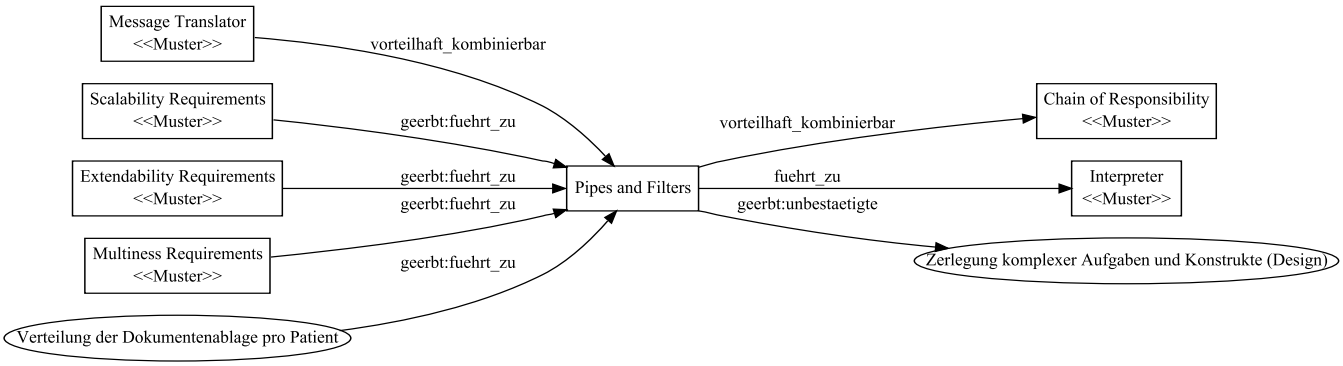
Gruppe(n)

- Grundlegende Architekturstile
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kurzbeschreibung

Das Pattern "Pipes and Filters" beschreibt eine Form der Verkettung bzw. Zerlegung der Bearbeitung mit kontinuierlicher Weitergabe der jeweiligen Zwischenergebnisse. Die in einer Reihe nacheinander verketteten Verarbeitungsschritte reichen ihre erreichten Ergebnisse jeweils als Eingangsinformation für den darauf folgenden Verarbeitungsschritt weiter.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Message Translator -> Pipes and Filters** : Verketten von Message Translators
  Quelle: Enterprise Integration Patterns; Hohpe; S. 89
- **Pipes and Filters -> Chain of Responsibility** : kann durch realisiert werden
  Das Pipes-and-Filters-Pattern teilt eine Aufgabe in eine Menge sequenzierbarer Arbeitsschritte. Diese Sequenz von Arbeitsschritten kann durch eine Chain of Responsibility abgebildet werden. Eine Chain of Responsibility besteht aus einer verketteten Liste von Handler-Objekten. Diese Handler-Objekte können verwendet werden um je einen Schritt aus dem Pipes-and-Filters-Architektur-Pattern abzubilden.
- **Pipes and Filters -> Interpreter** :
  Quelle: POSA1; S. 57;
  S. 57 zeigt, dass bei der typischen Anwendung der Pipes-and-Filters-Architektur Interpreter und Bestandteile von Interpretern verwendet werden.
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.
- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen

Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.

- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *138. Point to Point Messaging*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 103-105; als Point-to-Point Channel

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

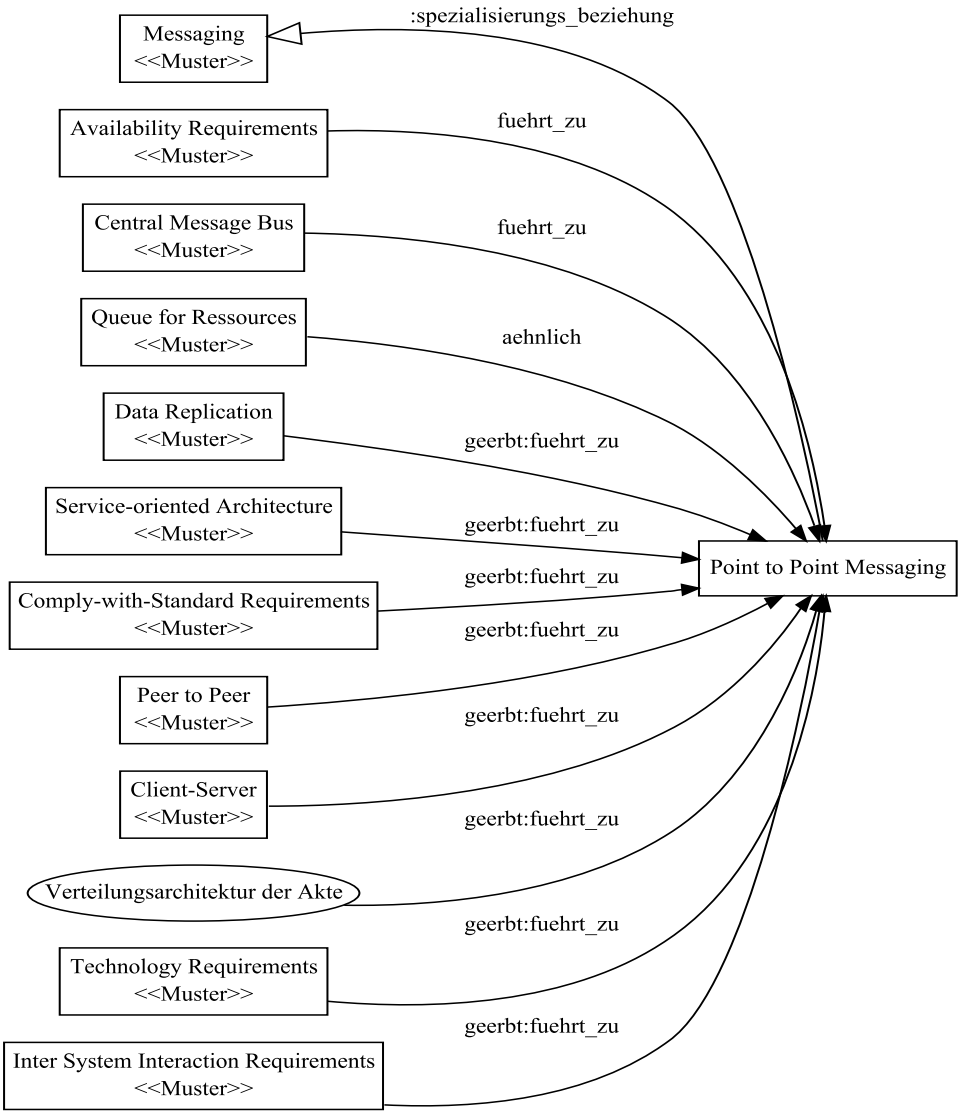
Gruppe(n)

- Asynchrone Kommunikation

Kurzbeschreibung

Das Pattern "Point to Point Messaging" beschreibt die asynchrone Übermittlung von Nachrichten zwischen zwei Kommunikanten mittels einer Nachrichten-Warteschlange. Dabei wird sichergestellt dass jede Nachricht nur von genau einem Empfänger verarbeitet wird.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Messaging -> Point to Point Messaging** : Spezialform
- **Availability Requirements -> Point to Point Messaging** : zuverlässige Nachrichtenübermittlung
  Durch die Anwendung von (transaktionalem) Point to Point Messaging über eine hochverfügbare Infrastruktur zur Nachrichtenübertragung können kürzere Ausfallzeiten einzelner am Gesamtsystem beteiligter Dienste oder Server kompensiert werden.
- **Central Message Bus -> Point to Point Messaging** : Unterstützt
  Point to Point Messaging beschreibt die zuverlässige Übertragung einer Nachricht von einem definierten Sender zu einem ebenfalls definierten Empfänger. Praktisch wird dieses Pattern häufig durch den Einsatz einer Message Queue (vgl. JMS) umgesetzt. Solchen Message Queues kann (siehe Enterprise Integration Patterns, Hohpe, S. 139) ein Message-Routing-Vorgang vorgeschalten sein. Der Central Message Bus kann als

zentrale Plattform für die zuverlässige Übermittlung von Nachrichten zwischen Sender und Empfänger dienen.

- **Data Replication -> Asynchrone Kommunikation** : verwendet
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion; „*There are many different ways to implement data replication. For example,* [...]*“* Alle der an dieser Stelle aufgeführten Methoden sind der asynchronen Kommunikation zuzuordnen.
- **Service-oriented Architecture -> Asynchrone Kommunikation** : für asynchrone Dienste
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: SOA in Practice, Josuttis; S. 125; Message Exchange Pattern: One-Way
  Quelle: SOA in Practice, Josuttis; S. 126; 10.2.3 Request/Response Versus Two One-Way Messages
  Quelle: SOA in Practice, Josuttis; S. 128; Message Exchange Pattern: Request/Callback
  Quelle: SOA in Practice, Josuttis; S. 129; Message Exchange Pattern: Publish/Subscribe
- **Comply-with-Standard Requirements -> Asynchrone Kommunikation** : wenn im Standard gefordert
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Peer to Peer -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Peer-to-Peer-Kommunikation ist mit den Mitteln asynchroner Kommunikation realisierbar.
- **Client-Server -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Client-Server-Systeme sind mit Mitteln asynchroner Kommunikation realisierbar
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Technology Requirements -> Asynchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.

- **Inter System Interaction Requirements -> Asynchrone Kommunikation** :
  Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.

## *139. Process Adaptivity Requirements*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

Kontext

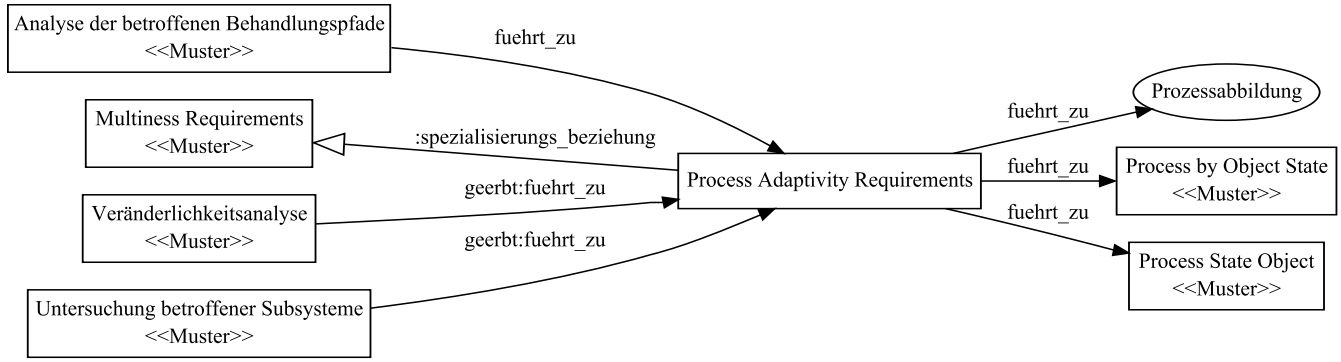
Prozesse und Verfahrensweisen ändern sich im Verlauf der Zeit aber auch Softwaresysteme werden von Zeit zu Zeit durch andere, modernere ersetzt. Besonders häufig veränderliche Prozesse stellen eine Herausforderung für die abbildenden Softwaresysteme dar.

Problem
Ein Softwaresystem mit der Fähigkeit auszustatten, Prozesse flexibel zu verändern führt zu einem deutlich erhöhten Entwicklungsaufwand. Fehlt diese Fähigkeit bei sich häufig ändernden Prozessen, so ist häufigere Neuprogrammierung notwendig. In beiden Fällen entstehen unnötige Kosten.

Lösung
Eine rechtzeitige Analyse von Veränderungspotenzialen in den abzubildenden Prozessen kann helfen, einen großen Teil dieser unnötigen Aufwände zu vermeiden. Eine Analyse der Veränderungspotenziale basiert auf der schrittweisen Analyse aller beteiligten Prozessschritte. Jeder der Prozessschritte wird nach folgenden Kriterien untersucht:

- Erfahrungswerte zu Änderungshäufigkeit (z. B. aus Altsystemen)
- Änderungshäufigkeit in relevanten Rechtsvorschriften
- Abhängigkeit von externen Faktoren wie z. B. Kooperationen mit anderen Einrichtungen

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der betroffenen Behandlungspfade -> Process Adaptivity Requirements** :
  Input: Pfade und letzte Änderungen an Pfaden
  Die Analyse der betroffenen Behandlungspfade liefert die relevanten Behandlungspfade (medizinische Prozesse) für die Process Adaptivity Requirements festgelegt werden müssen.
- **Multiness Requirements -> Process Adaptivity Requirements** : Vielfältigkeit der Prozesse
  Die an Prozesse gestellten Vielfältigkeitanfoderungen werden durch das Process-adaptivitiy-Requirements-Pattern beschrieben.
- **Process Adaptivity Requirements -> Prozessabbildung** : Input: Anforderungen
  Die Process Adaptivity Requirements sind die Flexibilitätsanforderungen an den gewählten Mechanismus zur Prozessabbildung. Hohe Flexibilitätsanforderungen benötigen umfassende Möglichkeiten zur Konfiguration von Prozessen, niedrige Anforderungen aus relativ statischen Prozessen führen zu stärker direkt im Quellcode des Systems realisierten Prozessen.
- **Process Adaptivity Requirements -> Process by Object State** : Bei geringer Veränderlichkeit
  Bei geringer Veränderlichkeit des Prozesses kann der Prozessstatus direkt in den Objekten des Domain-Models abgebildet werden.
- **Process Adaptivity Requirements -> Process State Object** : Bei größerer Veränderlichkeit
  Besteht die Wahrscheinlichkeit häufiger Veränderungen an Prozessen, so ist es vorteilhaft, die Verwaltung des Prozesszustands außerhalb des Domänenmodells zu betreiben.
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.

- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *140. Process by Object State*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Verwaltung des Prozesszustands

Kontext

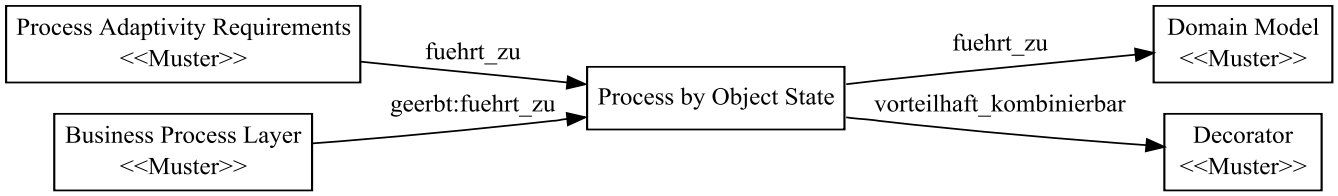
Zur elektronischen Unterstützung von Prozessen werden Dokumente und andere Objekte abhängig von ihrem Zustand unterschiedlichen Bearbeitern zugewiesen. Im Kontext einer verteilten Krankenakte kann ein solcher Prozess z. B. ein Behandlungspfad, also ein spezifisch für eine Art von Erkrankung festgelegter Behandlungsprozess sein. Besonders innerhalb medizinischer Einrichtungen legt der aktuelle Leistungserbringer, meist auf Basis der Ergebnisse der von ihm erbrachten Leistung, die im nächsten Schritt zuständige Leistungsstelle fest. Aber auch innerhalb von, aus Sicht des Behandlungspfads einzelnen Leistungen, kann die Leistungserbringung auf mehrere Akteure verteilt und durch einen elektronisch abgebildeten Workflow unterstützt sein (z. B. Röntgenworkflow).

Problem
Um eine systemseitige Unterstützung für definierte Prozesse bieten zu können, ist es notwendig, dass das System eigenständig den Zustand seiner Prozessinstanzen ermitteln kann.

Lösung
Der Zustand der Prozessinstanz wird aus dem Zustand, also den Attributwerten der beteiligten Objekte des Domänenmodellsl ermittelt. Zur Ermittlung des Zustands werden Methoden in den entsprechenden Klassen des Domänenmodells benötigt.

Beispiel
Quelle der Idee für dieses Pattern: SOA in Practice; Josuttis; S. 72; 6.4.2 Service State versus Backend State
Dieses Pattern entspricht dem in der Quelle beschriebenen Backend State, also einem im Backend verwalteten Prozesszustand.

Beziehungen - Grafik




Beziehungen - Detail

- **Process Adaptivity Requirements -> Process by Object State** : Bei geringer Veränderlichkeit
  Bei geringer Veränderlichkeit des Prozesses kann der Prozessstatus direkt in den Objekten des Domain-Models abgebildet werden.
- **Process by Object State -> Domain Model** : Prozesszustand als Teil des Domänenmodells
  Der Prozesszustand wird als Teil des Domänenmodells direkt in den verschiedenen Entitäten mit abgebildet. Das kann auch eine Berechnung des Prozesszustands aus den Zuständen verschiedener Attribute bedeuten.
- **Process by Object State -> Decorator** : Zur flexibleren Gestaltung der Zustandsermittlung
  Wird der Prozesszustand aus den Attributen des Domänenmodells ermittelt, so kann, besonders wenn das System verschiedene Prozesse abbildet, durch die Verwendung des Decorator-Patterns das Domain-Model ohne die zusätzliche Komplexität der Ermittlung des Prozesszustands umgesetzt werden.
- **Business Process Layer -> Verwaltung des Prozesszustands** : notwendig zur Abbildung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen Geschäftsprozess durch Software abbilden zu können, muss das zugehörige IT-System Kenntniss über den Zustand einer jeden Prozess-Instanz besitzen. Auf Basis des Zustands wird vom System der nächste notwendige Bearbeitungsschritt ermittelt. Wie dieser Zustand technisch verwaltet wird kann unterschiedlich gelöst werden. Die Gruppe Verwaltung des Prozesszustands stellt mehrere Kandidaten zur Auswahl.

## *141. Process State Object*

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Verwaltung des Prozesszustands

## Kontext

Zur elektronischen Unterstützung von Prozessen werden Dokumente und andere Objekte abhängig von ihrem Zustand unterschiedlichen Bearbeitern zugewiesen. Im Kontext einer verteilten Krankenakte kann ein solcher Prozess z. B. ein Behandlungspfad, also ein spezifisch für eine Art von Erkrankung festgelegter Behandlungsprozess sein. Besonders innerhalb medizinischer Einrichtungen legt der aktuelle Leistungserbringer, meist auf Basis der Ergebnisse der von ihm erbrachten Leistung, die im nächsten Schritt zuständige Leistungsstelle fest. Aber auch innerhalb von, aus Sicht des Behandlungspfads einzelnen Leistungen, kann die Leistungserbringung auf mehrere Akteure verteilt und durch einen elektronisch abgebildeten Workflow unterstützt sein (z. B. Röntgenworkflow).

## Problem

Um eine systemseitige Unterstützung für definierte Prozesse bieten zu können, ist es notwendig, dass das System eigenständig den Zustand seiner Prozessinstanzen ermitteln kann. Zusätzlich besteht eine hohe Wahrscheinlichkeit von häufigen Änderungen an den abzubildenden Prozessen. Eine Verlagerung des Prozesszustands in das Domänenmodell würde somit zu häufigen Änderungen am Domänenmodell führen.

## Lösung

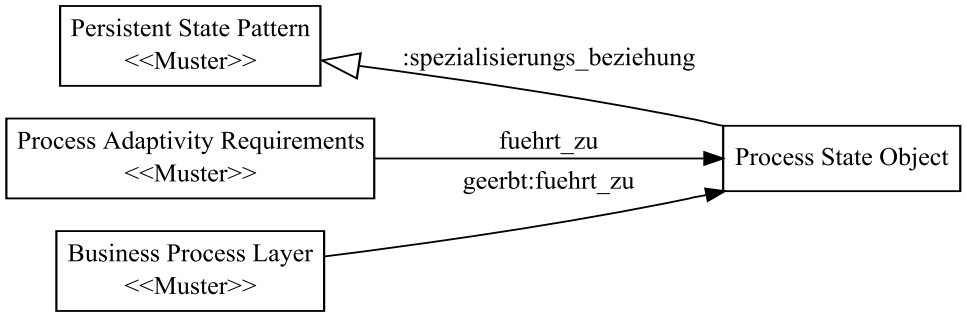
Der Status der Prozessinstanzen wir durch eine zusätzliche persistente Klasse pro Prozesstyp abgebildet. Hilfreich ist dabei ein Aufzählungstyp (Enumeration), der Werte für alle gültigen Zustände enthält. Die Klasse zur Prozessinstanz enthält mindestens folgende Attribute:

- Eine eindeutige Nummer zur Identifizierung der Prozessinstanz,
- die ID des mit der Prozessinstanz verbundenen Objekts und
- entweder eine Liste von Elementen des Aufzählungstyps zur Historisierung bisher durchgeführter Prozesschritte
- oder ein einzelnes Attribut des Aufzählungstyps das den aktuellen Prozesszustand wiedergibt.

## Beispiel

Quelle der Idee für dieses Pattern: SOA in Practice; Josuttis; S. 72; 6.4.2 Service State versus Backend State

## Beziehungen - Grafik



Beziehungen - Detail

- **Persistent State Pattern -> Process State Object** :
  Das Process State Object ist eine Form des Persistent-State-Pattern.
- **Process Adaptivity Requirements -> Process State Object** : Bei größerer Veränderlichkeit
  Besteht die Wahrscheinlichkeit häufiger Veränderungen an Prozessen, so ist es vorteilhaft, die Verwaltung des Prozesszustands außerhalb des Domänenmodells zu betreiben.
- **Business Process Layer -> Verwaltung des Prozesszustands** : notwendig zur Abbildung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen Geschäftsprozess durch Software abbilden zu können, muss das zugehörige IT-System Kenntniss über den Zustand einer jeden Prozess-Instanz besitzen. Auf Basis des Zustands wird vom System der nächste notwendige Bearbeitungsschritt ermittelt. Wie dieser Zustand technisch verwaltet wird kann unterschiedlich gelöst werden. Die Gruppe Verwaltung des Prozesszustands stellt mehrere Kandidaten zur Auswahl.

## *142. Protection Reverse Proxy*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 457-464

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

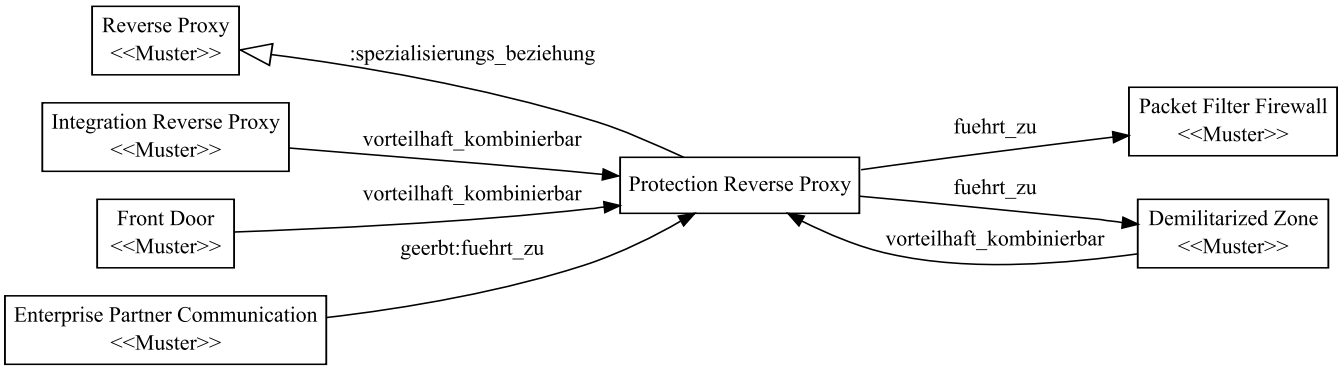
Gruppe(n)

- Access Restriction

Kurzbeschreibung

Das Pattern "Protection Reverse Proxy" beschreibt den Ansatz, einen Reverse Proxy zum Ausfiltern potenziell gefährlicher Anfragen zu verwenden, um so den (oder die) eigentlichen Server vor bestimmten Arten von Angriffen zu schützen.

Beziehungen - Grafik




Beziehungen - Detail

- **Reverse Proxy -> Protection Reverse Proxy** : Ist eine Spezialform von
  Der Protection Reverse Proxy ist ein spezieller Reverse Proxy.
- **Demilitarized Zone -> Protection Reverse Proxy** : Anbindung dynamischer Inhalte
  Quelle: S. 451; Security Patterns; Schumacher et al.
- **Integration Reverse Proxy -> Protection Reverse Proxy** : Integration Protection Reverse Proxy
  Quelle: S. 470; Security Patterns; Schumacher et al.
  *"**Variants***
  *Integration Protection Reverse Proxy. It is easy (and wise) to combine the Integration Reverse Proxy with the Protection Reverse Proxy and gain the benefits of both."*
- **Front Door -> Protection Reverse Proxy** :
  Quelle: Security Patterns; Schumacher et al.; S. 462;
  *"Integration Reverse Proxy and Front Door can(and should) be combined in their function with Protection Reverse Proxy, and thus vary this pattern by adding functionality."*
- **Protection Reverse Proxy -> Packet Filter Firewall** :
  Quelle: Security Patterns; Schumacher et al.; S. 458;
  ***Solution***
  "[...] *Two Packet Filter Firewalls ensure that no external network traffic reaches the real Web server."*
- **Protection Reverse Proxy -> Demilitarized Zone** :
  Quelle: Security Patterns; Schumacher et al.; S. 459;
  "[...] *The resulting network topology provides a Demilitarized Zone containing only the reverse proxy machine, and a secured server zone containing the Web server."*
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *143. Protective Automatic Controls*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 198-200

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

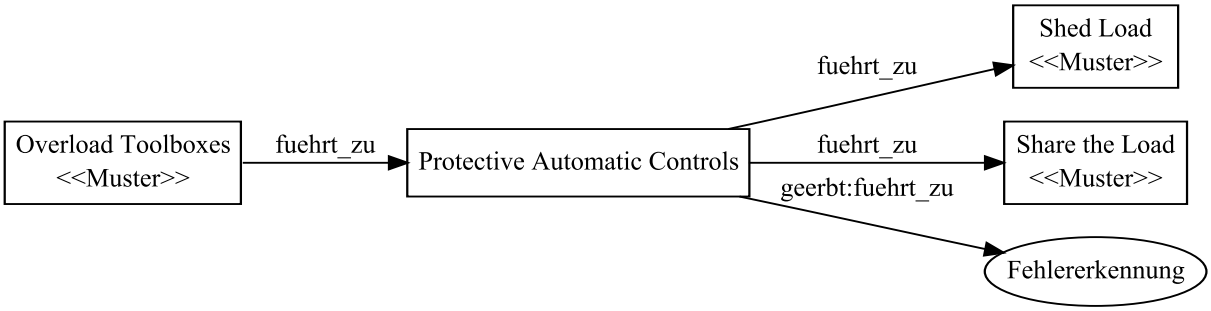
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Protective Automatic Controls" befasst sich mit automatischen Mechanismen, die die Menge der Anfragen, die ein System annimmt dann beschränken, wenn erkennbar ist, dass eine weitere Steigerung zu Leistungseinbrüchen durch interne Aufgaben, wie z. B. die Zuteilung des Zugriffs auf knappe Ressourcen, führen kann.

Beziehungen - Grafik




Beziehungen - Detail

- **Overload Toolboxes -> Protective Automatic Controls** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Protective Automatic Controls -> Shed Load** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Protective Automatic Controls -> Share the Load** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *144. Prototype*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 117-126

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

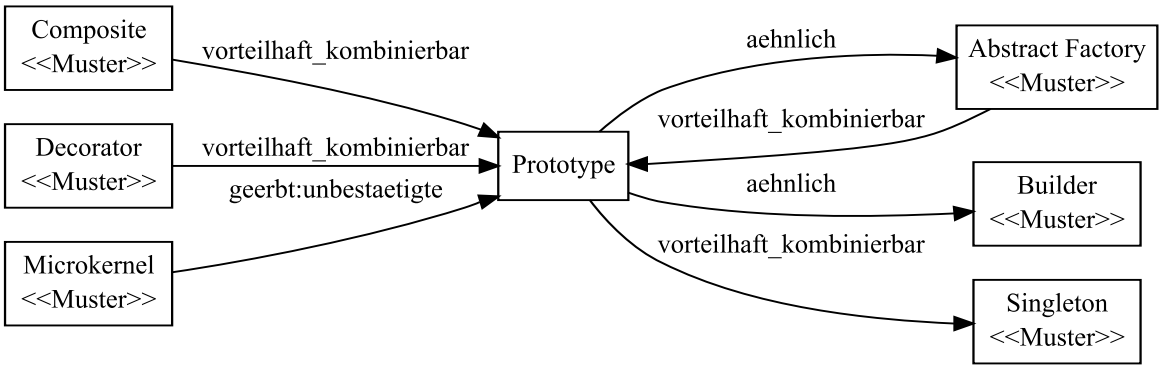
Gruppe(n)

- Instanziierung und Kontrolle der Instanzen

Kurzbeschreibung

Das Pattern "Prototype" beschreibt den Ansatz, häufig gebrauchte Objekte mit ihren Attributwerten als Prototypen vorzubereiten, um sie dann, wenn ein neues Objekt gleicher Art benötigt wird, durch einfaches Clonen vermehren zu können.

Beziehungen - Grafik




Beziehungen - Detail

- **Abstract Factory -> Prototype** : Configure factory dynamically
  Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Composite -> Prototype** :
  Quelle: Design Patterns; Gamma et al.; S. 126;

*"Designs that make heavy use of Composite and Decorator patterns often can benefit from Prototype as well."*

- **Decorator -> Prototype** :
  Quelle: Design Patterns; Gamma et al.; S. 126;
  *"Designs that make heavy use of Composite and Decorator patterns often can benefit from Prototype as well."*
- **Prototype -> Abstract Factory** :
  Quelle: Design Patterns; Gamma et al.; S. 119;
  *"Prototype has many of the same consequences that Abstract Factory and Builder have: "*
- **Prototype -> Builder** :
  Quelle: Design Patterns; Gamma et al.; S. 119;
  *"Prototype has many of the same consequences that Abstract Factory and Builder have: "*
- **Prototype -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 134
  *"Many patterns can be implemented using the Singleton pattern. See Abstract Factory (87), Builder(97), and Prototype (117)."*
- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen
  Geerbte Beziehung: unbestaetigte_beziehung
  Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.
  Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

## *145. Proxy*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 207-217

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

**POSA 1, A System of Patterns, Buschmann**

S. 263-275

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Flexibilität

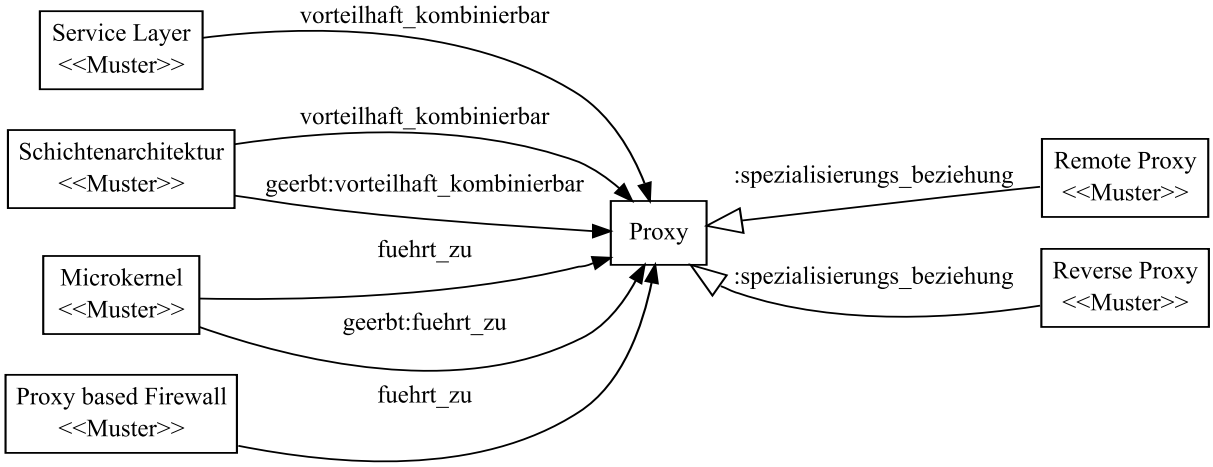
### Gruppe(n)

- Kopplung und Entkopplung

### Kurzbeschreibung

Das Proxy-Pattern ist die allgemeinste Form des Stellvertreter Patterns. Es beschreibt die Verwendung einer Stellvertreter-Klasse, die somit den nutzenden Code von der Implementierung der genutzten Klasse entkoppelt.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Service Layer -> Proxy** :
  Um die Kopplung zwischen der konkreten Implementierung der Dienste der Service-Layer und den konsumierenden Softwarebestandteilen der darüberliegenden Schicht zu verringern, kann die Schnittstelle der Service Layer durch Stellvertreter-Objekte (Proxys) gekapselt werden.
- **Schichtenarchitektur -> Proxy** :
  Generell kann die Kopplung zwischen den Schichten einer Schichtenarchitektur durch die Kappselung der Schnittstelle der jeweils konsumierten Schicht mit Stellvertreterobjekten (Proxys) verringert werden. Werden die Schichten der Architektur in einem verteilten Szenario betrieben, so ist die Verwendung von Remote Proxys zwischen den Schichten sogar notwendig um die Kommunikation zu verbergen.
- **Microkernel -> Proxy** : Implementierung der Adaptoren
  Quelle: POSA 1; Buschmann et al.; Abhängigkeitsdiagramm auf der letzten Seite (ohne Nummer);
  *"Microkernel: implementing adapters"* auf dem Pfeil der Beziehung zwischen Broker, Microkernel und Proxy.
- **Proxy based Firewall -> Proxy** : Bestandteil
  Quelle: Security Patterns; Schumacher et al.; S. 416

*"This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall."*

- **Proxy -> Remote Proxy** :
  Eine Spezialform des Proxy ist der Remote Proxy. Er dient der Kapselung entfernter Kommunikation in einem lokalen, dem entfernten Objekt bezüglich seiner Schnittstelle äquivalenten Stellvertreter-Objekt.
- **Proxy -> Reverse Proxy** :
  Eine Spezialform des Proxy ist der Reverse Proxy. Beim Reverse Proxy handelt es sich üblicherweise nicht um einen Stellvertreter für ein Objekt, sondern um einen Stellvertreter für einen oder mehrere Server. Der Reverse Proxy macht deren Inhalte in einem Netz verfügbar, aus dem man diese Server nicht direkt erreichen kann.
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *146. Proxy based Firewall*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 411-416

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

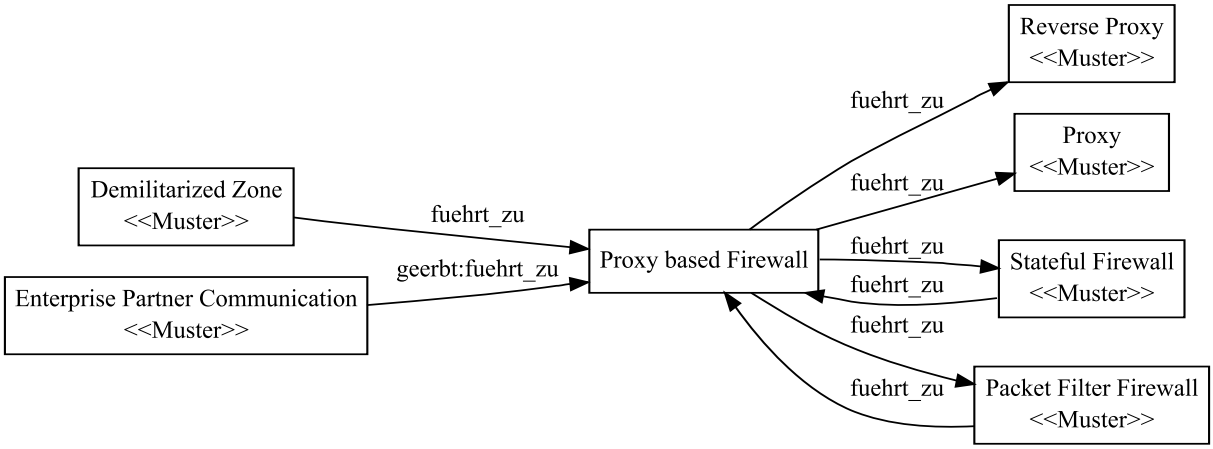
Gruppe(n)

- Access Restriction

## Kurzbeschreibung

Das Pattern "Proxy based Firewall" beschreibt einen Ansatz, der auf einem Stellvertreter-Server aufbaut, der die durch ihn übertragenen Anfragen und Antworten auf Basis eines gegebenen Regelwerks filtert.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Demilitarized Zone -> Proxy based Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Packet Filter Firewall -> Proxy based Firewall** : Wenn Firewall auf Applikationsebene benötigt
  Wenn die Verwendung einer Packet Filter Firewall nicht ausreichend ist, weil die Bedrohung auf Angriffen basiert, die auf einer höheren Protokollebene stattfinden, kann ggf. mit einer Proxy based Firewall abhilfe geschaffen werden.
  Quelle: Security Patterns; Schumacher et al.; S. 411
- **Stateful Firewall -> Proxy based Firewall** : Üblicherweise kombiniert
  Quelle: Security Patterns; Schumacher et al; S. 421
  *"This firewall* [Stateful Firewall] *is usually combined with one or both of the previous types of firewalls, Packet Filter Firewall and Proxy-Based Firewall."*
- **Proxy based Firewall -> Reverse Proxy** : Im verteilten Szenario
  Zur Umsetzung einer Proxy based Firewall im verteilten Szenario ist ein Reverse Proxy empfehlenswert.
  Quelle: Security Patterns; Schumacher et al.; S.414
  *"Consider configurations such as Protection Reverse Proxy, Integration Reverse Proxy or a combination with a Packet Filter Firewal in a distributed configuration"*
- **Proxy based Firewall -> Proxy** : Bestandteil
  Quelle: Security Patterns; Schumacher et al.; S. 416
  *"This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall."*
- **Proxy based Firewall -> Stateful Firewall** : Zum Schutz gegen zusätzliche Angriffsformen

Quelle: Security Patterns; Schumacher et al.; S.416
*"This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall."*
Quelle: Security Patterns; Schumacher et al.; S.417
*"We have been able to contain many attacks with Packet Filter Firewall and Proxy-Based Firewall. However, we are still plagued with distributed denial of service attacks that prevent customers from reaching our site. We also have* […]*"*

- **Proxy based Firewall -> Packet Filter Firewall** : Filterung auf Netzwerkebene
  Quelle: Security Patterns; Schumacher et al.; S.416
  *"This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall."*
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *147. Publish-Subscribe Messaging*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 106-110; als Publish-Subscribe Channel

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

**POSA 1, A System of Patterns, Buschmann**

S. 339 ff.; Grundprinzip Publisher-Subscriber; ohne speziellen Bezug zu Messaging

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Kommunikation

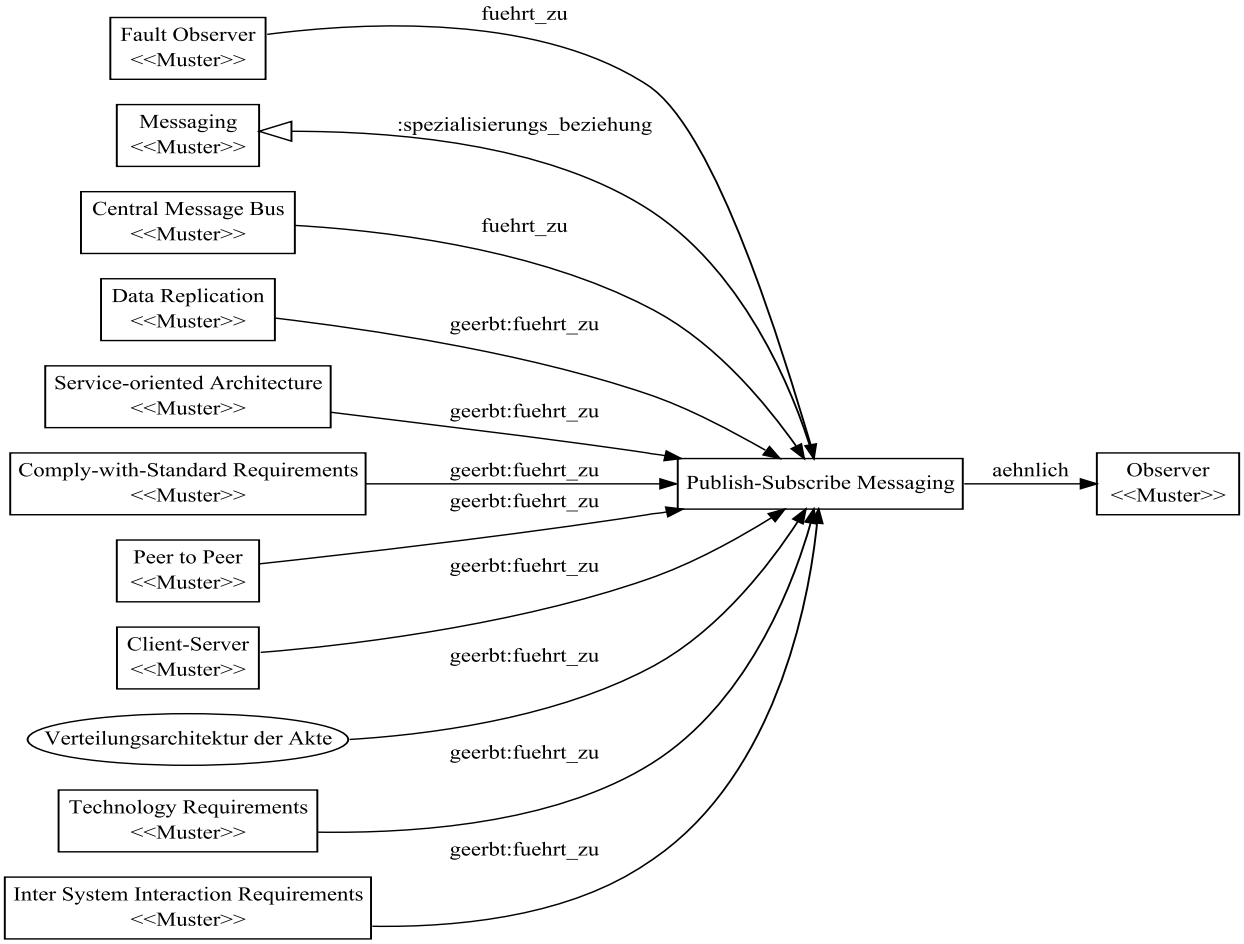
Gruppe(n)

- Asynchrone Kommunikation

Kurzbeschreibung

Das Pattern "Publisch-Subscribe Messaging" beschreibt einen dem Observer-Pattern ähnlichen Ansatz, der allerdings durch die asynchrone Übermittlung von Nachrichten umgesetzt ist. Komponenten können sich für den Empfang bestimmter Nachrichten registrieren und erhalten diese, sofern sie verfügbar sind, asynchron zugestellt.

Beziehungen - Grafik




Beziehungen - Detail

- **Fault Observer -> Publish-Subscribe Messaging** : Bestandteil der Umsetzung
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 74
  *"The Publischer-Subscriber pattern describes an effective way of distributing this fault information."*
- **Messaging -> Publish-Subscribe Messaging** : Spezialform
- **Central Message Bus -> Publish-Subscribe Messaging** : Unterstützt
  Quelle: Enterprise Integration Patterns, Hohpe, S. 139
- **Publish-Subscribe Messaging -> Observer** :
  Das Observer-Pattern und das Publisher-Subscriber-Pattern, das speziell den Anwendungsfall Messaging des Observer-Patterns beschreibt, sind bezogen auf ihre Auswirkung ähnlich.
- **Data Replication -> Asynchrone Kommunikation** : verwendet
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 7; Enterprise Integration Patterns; kurz vor Diagramm für Shared Business Funktion; „*There are many different ways to implement data replication. For example,* [...]*"* Alle der an dieser Stelle aufgeführten Methoden sind der asynchronen Kommunikation zuzuordnen.

- **Service-oriented Architecture -> Asynchrone Kommunikation** : für asynchrone Dienste
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: SOA in Practice, Josuttis; S. 125; Message Exchange Pattern: One-Way
  Quelle: SOA in Practice, Josuttis; S. 126; 10.2.3 Request/Response Versus Two One-Way Messages
  Quelle: SOA in Practice, Josuttis; S. 128; Message Exchange Pattern: Request/Callback
  Quelle: SOA in Practice, Josuttis; S. 129; Message Exchange Pattern: Publish/Subscribe
- **Comply-with-Standard Requirements -> Asynchrone Kommunikation** : wenn im Standard gefordert
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Peer to Peer -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Peer-to-Peer-Kommunikation ist mit den Mitteln asynchroner Kommunikation realisierbar.
- **Client-Server -> Asynchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Client-Server-Systeme sind mit Mitteln asynchroner Kommunikation realisierbar
- **Verteilungsarchitektur der Akte -> Asynchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Technology Requirements -> Asynchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Inter System Interaction Requirements -> Asynchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.

## 148. Pull-Prinzip

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Richtung des Datenaustauschs

Quellen

Peter Rechenberg (Herausg.); Informatik-Handbuch; 4. Auflage; Hanser Verlag; 2006; S. 728 - 731

Abweichend dazu: Pull-Prinzip in der Logistik, vgl. z. B. Hans-Otto Günther, Horst Tempelmeier; Produktion und Logistik; 6. Auflage; Springer; 2005; S. 315 ff.

Kontext

Bei der Konzeption eines IT-Systems zur Abbildung einer verteilten Krankenakte muss zu Beginn der Konzeption der Systemarchitektur entschieden werden, ob Informationen zum Zeitpunkt ihrer Erstellung aktiv an potenzielle Nutzungsorte verteilt oder zum Zeitpunkt der Nutzung aktiv am Entstehungsort abgeholt werden sollen.

Problem

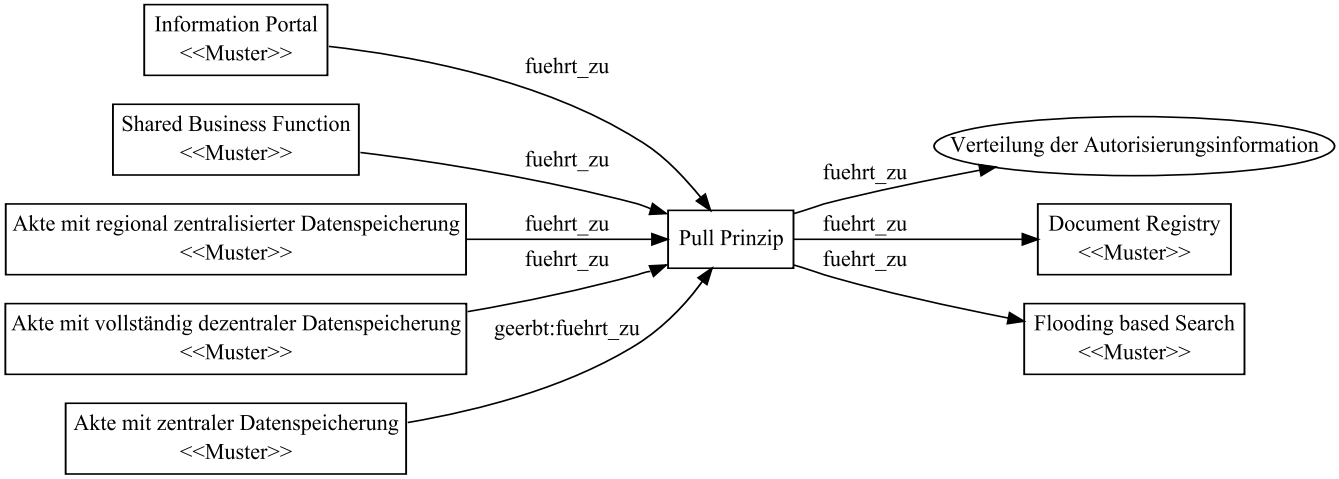
Zum Zeitpunkt der Entstehung eines Dokuments oder sonstigen Inhalts ist noch nicht abschließend bekannt wo und von wem es später genutzt werden soll. Außerdem macht eine aktive Verteilung an alle beteiligten Knoten es unmöglich, dass ein Patient seine Einwilligung zur Weitergabe nachträglich zurücknimmt. Außerdem steht sie grundsätzlich im Konflikt zum Prinzip der Datensparsamkeit.

Lösung

Das System wird so konzipiert, dass die Daten am Ort ihrer Entstehung, einem zentralen Ort oder einem anderen speziell dafür vorgesehenen Ort gespeichert werden. Subsysteme die Daten benötigen, die nicht in Ihnen selbst gespeichert sind, holen diese aktiv an der Stelle der Datenspeicherung ab.

Dieses Verfahren setzt voraus, dass jedes Subsystem die Möglichkeit hat, Daten die in einem anderen Subsystem existieren aufzufinden.

Beziehungen - Grafik




Beziehungen - Detail

- **Information Portal -> Pull Prinzip** : basiert auf Pull
  Das Information Portal holt Daten aktiv (Pull) aus den Quellsystemen um sie im Portal anzuzeigen.
- **Shared Business Function -> Pull Prinzip** : Übliches Kommunikationsprinzip
  Anwendungen, die auf Shared Business Functions aufbauen, sind bezogen auf ihre hauptsächliche Kommunikationsrichtung häufig nach dem Pull-Prinzip aufgebaut. Das bedeutet, dass Daten nicht an alle später möglichen Konsumenten verteilt werden, sondern über eine gemeinsame Funktion (Shared Business Function) bei Bedarf abgerufen werden.
- **Akte mit regional zentralisierter Datenspeicherung -> Pull Prinzip** : Datenzugriff
  Der Datenzugriff auf die Inhalte von Akten mit regional zentralisierter Datenspeicherung kann vorteilhaft mittels des Pull-Prinzips (umgesetzt z. B. durch Remote Procedure Invocation) erfolgen.
- **Akte mit vollständig dezentraler Datenspeicherung -> Pull Prinzip** : Ansatz 2: Zugriff auf Inhalte
  Wird der Ansatz 2 einer Akte mit vollständig dezentraler Datenspeicherung realisiert, so folgt sämtliche Kommunikation dem Pull-Prinzip. Berechtigte Benutzer greifen durch gezielte Anfragen von entfernten Systemen auf enthaltene Dokumente zu. Dies entspricht aktivem Abholen bei Bedarf (Pull).
- **Pull Prinzip -> Aktenbezogene Autorisierungskonzepte** : Gewährleistung des Zugriffsschutzes
  Erfolgt der Transport der Akteninhalte nach dem Pull-Prinzip, so ist die verteilte oder zentrale Verwaltung von Autorisierungsregeln zwangsläufig notwendig um beim Zugriff den Schutz vor unberechtigtem Zugriff zu gewährleisten.
- **Pull Prinzip -> Document Registry** : Zur Findung abholbarer Dokumente
  Bei der Verwendung des Pull-Prinzips kann eine zentrale Document Registry notwendig (oder zumindest hilfreich) sein, um den Ablageort eines Dokuments zu finden. Die Kenntnis über den Ablageort ist notwendig, um das Dokument aktiv abholen zu können.

- **Pull Prinzip -> Flooding based Search** : Zur Findung abholbarer Dokumente
  Alternativ zur Verwendung einer zentralen Document Registry kann der Ablageort eines Dokuments auch über einen Flooding-based-Search-Mechanismus ermittelt werden. Dabei werden alle bekannten Ablageorte rekursiv nach Dokumenten oder spezifischen Suchkriterien durchsucht. Die dadurch erlangte Kenntnis über den Ablageort ist notwendig, um das Dokument aktiv abholen zu können.
- **Akte mit zentraler Datenspeicherung -> Richtung des Datenaustauschs** :
  Situationsabhängige Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei einer Akte mit zentraler Datenspeicherung können die beiden Prinzipien Push und Pull einander ergänzend verwendet werden. Je nach gewähltem Lösungsansatz wird die Kombination beider Prinzipien etwas unterschiedlich gestaltet. Grundsätzlich werden Daten nach der Erstellung an den zentralen Speicherort übertragen oder durch einzelne Übertragungen von Teilen Stückweise dort erstellt. Der lesende bzw. suchende Zugriff auf die Daten erfolgt in beiden Fällen via Pull.

## *149. Push-Prinzip*

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Richtung des Datenaustauschs

Quellen

Peter Rechenberg (Herausg.); Informatik-Handbuch; 4. Auflage; Hanser Verlag; 2006; S. 728 - 731

Kontext

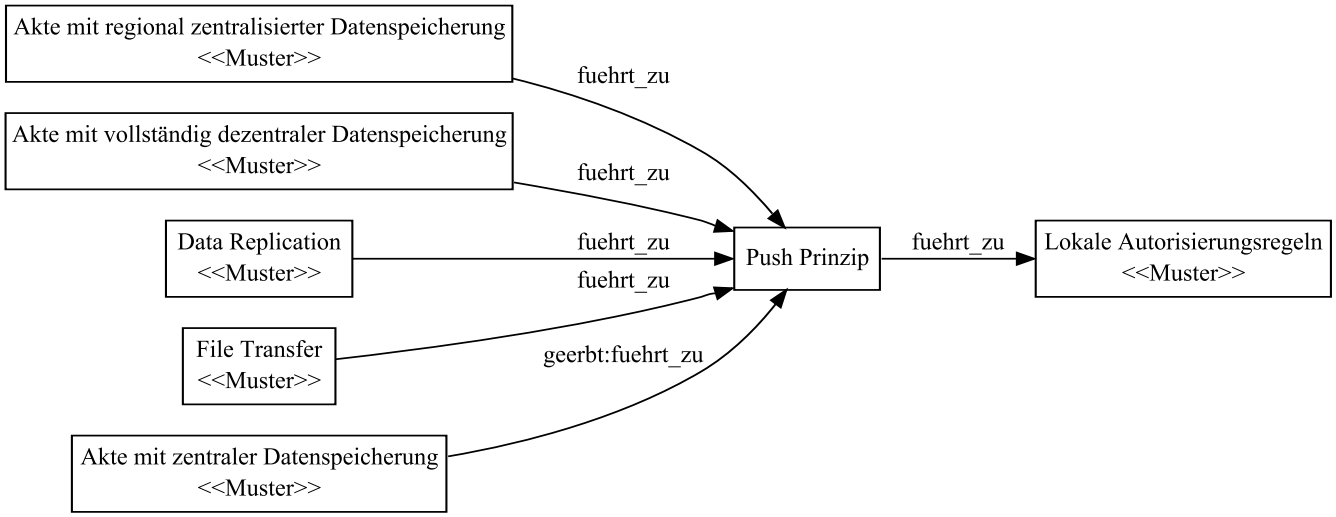
Bei der Konzeption eines IT-Systems zur Abbildung einer verteilten Krankenakte muss zu Beginn der Konzeption der Systemarchitektur entschieden werden, ob Informationen zum Zeitpunkt ihrer Erstellung aktiv an potenzielle Nutzungsorte verteilt oder zum Zeitpunkt der Nutzung aktiv am Entstehungsort abgeholt werden sollen.

Problem
Das nachträgliche Abholen von Dokumenten an ihrem Entstehungsort erfordert Zeit. Außerdem ist eine komplexe Berechtigungssystematik notwendig um die Anforderungen des Datenschutzes zu erfüllen.

Lösung
Der Systemknoten, in dem die Daten erzeugt oder verändert werden benachrichtigt alle anderen Systemknoten über die Änderung, indem er ihnen die geänderten Daten übermittelt. Dabei können die genannten anderen Systemknoten sowohl die gesamte Menge der im

System bekannten Knoten, als auch eine für den spezifischen Anwendungsfall relevante Teilmenge der im System verfügbaren Knoten sein. Eine typische Teilmenge ist bei verteilten Krankenakten die Menge der Systemknoten, die von Benutzern genutzt werden, die aufgrund definierter Behandlungspfade als voraussichtlich zukünftige Behandler eines Patienten identifizierbar sind.

Beziehungen - Grafik




Beziehungen - Detail

- **Akte mit regional zentralisierter Datenspeicherung -> Push Prinzip** : Synchronisation
  Das Push-Prinzip (umgesetzt Beispielsweise durch Point-to-Point-Messaging oder Publish-Subscribe-Messaging) kann in der Architektur einer Akte mit regional zentralisierter Datenspeicherung dazu genutzt werden um z. B. Stammdaten zwischen den regionalen Speicherknoten zu synchronisieren.
- **Akte mit vollständig dezentraler Datenspeicherung -> Push Prinzip** : Ansatz 1: Synchronisation
  Bei verteilten Krankenakten mit vollständig dezentraler Datenspeicherung die den Ansatz 1 des Patterns oder eine abgewandelte Form davon verwenden, folgt die Kommunikation dem Push Prinzip. Die Architektur der Akte erfordert eine vollständige Synchronisation der Stammdaten zwischen allen Knoten, so dass Änderungen immer an alle Knoten propagiert werden müssen. Auch Dokumente werden nach dem Push-Prinzip an alle als berechtigt identifizierten Empfänger versendet.
- **Data Replication -> Push Prinzip** : Basiert auf Push
  Änderungen werden zwecks Replikation aktiv an die Datensenken übermittelt. Dieses Vorgehen ist notwendig, da nur das System, in dem ein Datensatz geändert wird, von der Änderung Kenntnis hat. Aktives Nachfragen (Pull) durch die Systeme, die ein Replikat der Änderung erhalten sollen, wäre nur durch Polling (regelmäßiges Nachfragen ohne inhaltlich bedingten Auslöser) möglich.

- **File Transfer -> Push Prinzip** :
  Datenübertragung per File-Transfer kann nur als aktives Ausliefern vom Sender an den oder die Empfänger (Push-Prinzip) realisiert werden.
- **Push Prinzip -> Lokale Autorisierungsregeln** : zwangsläufig
  Werden die Daten nach dem Push-Prinzip an die verschiedenen (jeweils eigenständigen) Systemknoten verteilt, kann der Zugriff auf diese Daten auch ausschließlich über deren lokale Autorisierungsregeln gesteuert werden.
- **Akte mit zentraler Datenspeicherung -> Richtung des Datenaustauschs** :
  Situationsabhängige Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei einer Akte mit zentraler Datenspeicherung können die beiden Prinzipien Push und Pull einander ergänzend verwendet werden. Je nach gewähltem Lösungsansatz wird die Kombination beider Prinzipien etwas unterschiedlich gestaltet. Grundsätzlich werden Daten nach der Erstellung an den zentralen Speicherort übertragen oder durch einzelne Übertragungen von Teilen Stückweise dort erstellt. Der lesende bzw. suchende Zugriff auf die Daten erfolgt in beiden Fällen via Pull.

## *150. Quarantine*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 143-144

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

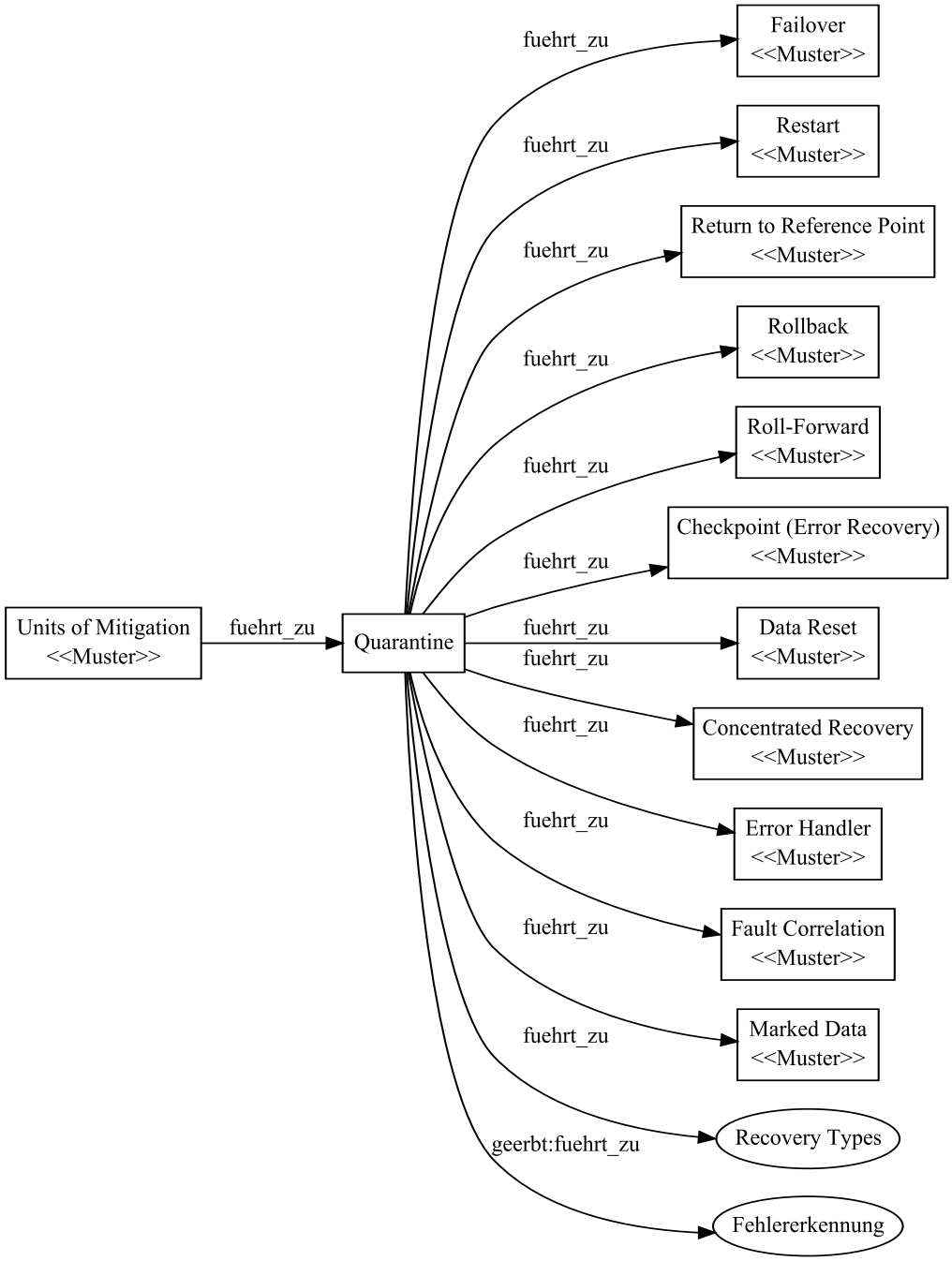
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Quarantine" beschreibt den Ansatz, fehlerhafte Daten oder Anfragen, die zu fehlerhafter Verarbeitung führen in einen Quarantäne-Bereich zu verschieben, um ihn z. B. für die manuelle Korrektur zugängig zu machen.

Beziehungen - Grafik




Beziehungen - Detail

- **Units of Mitigation -> Quarantine** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Quarantine -> Failover** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Restart** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
  **Hinweis:** Restart ist eine Lösung, die nur zur Kompensation von Fehlern bei transaktionalen, idempotenten und zustandsinvarianten Diensten verwendet werden

kann. Bei Diensten, Methoden oder Funktionen, die diese Eigenschaften nicht erfüllen kann die wiederholte Ausführung zur Produktion fehlerhafter Daten führen.

- **Quarantine -> Return to Reference Point** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Rollback** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Roll-Forward** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Checkpoint (Error recovery)** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Data Reset** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141 Abbildung 47
- **Quarantine -> Concentrated Recovery** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Quarantine -> Error Handler** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Quarantine -> Fault Correlation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Quarantine -> Marked Data** :
  Quelle: Letzte Seite, A Pattern Language for Fault Tolerant Software; Patterns for Fault Tolerant Software; Robert S. Hanmer
- **Quarantine -> Recovery Types** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *151. Queue for Ressources*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 193-194

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

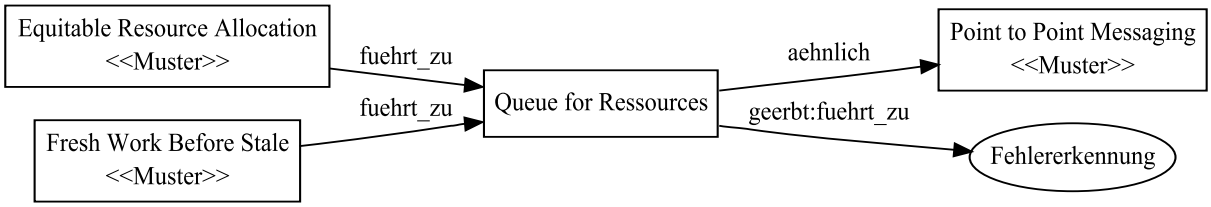
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Queue for Ressources" befasst sich mit der Frage, wie mit Anfrage verfahren werden soll, die zu dem Zeitpunkt, zu dem sie am System eintreffen nicht direkt verarbeitet werden können. Es schlägt vor, solche Anfragen in eine Warteschlange fester Länge einzureihen, um das Auftreten von Verfügbarkeitsfehlern zu vermeiden.

Beziehungen - Grafik




Beziehungen - Detail

- **Equitable Resource Allocation -> Queue for Ressources** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fresh Work Before Stale -> Queue for Ressources** : Als „Last In First Out“-Queue
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; S. 217-219
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *152. Realisitc Threshold*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 110-117

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

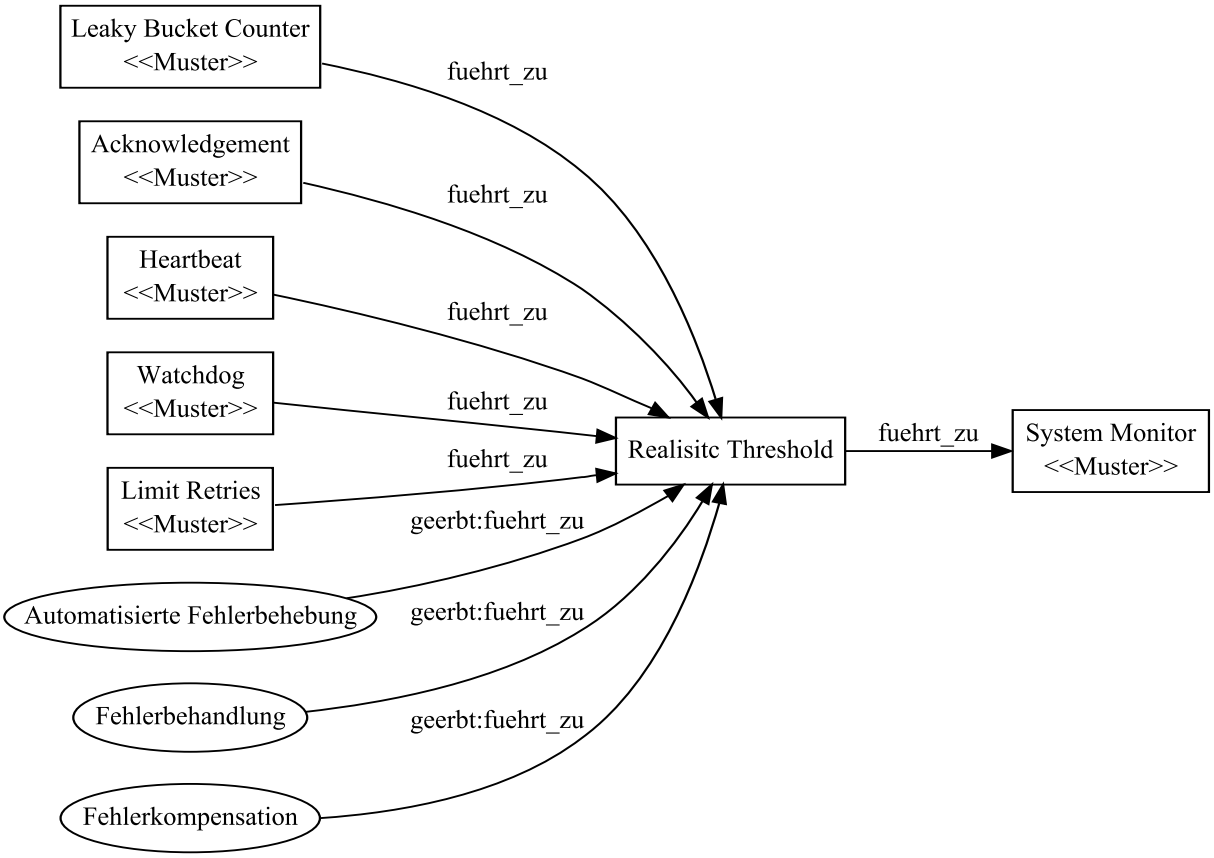
Gruppe(n)

- Fehlererkennung

## Kurzbeschreibung

Das Pattern "Realistic Threshold" beschreibt die Notwendigkeit bei der Erkennung von Fehlern auf entfernten Knoten entsprechende Latenz-Zeiten einzuplanen, um nicht aufgrund der Dauer der Nachrichtenübermittlung unbegründete Fehlermeldungen zu erhalten.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Leaky Bucket Counter -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Acknowledgement -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Heartbeat -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; S. 87
- **Watchdog -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Limit Retries -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Realisitc Threshold -> System Monitor** : Wenn Fehler erkannt wurde
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 114
  *“Refer to System Monitor for a discussion of what steps it can take when the error is detected.”*

- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *153. Reassess Overload Decision*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 189-190

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

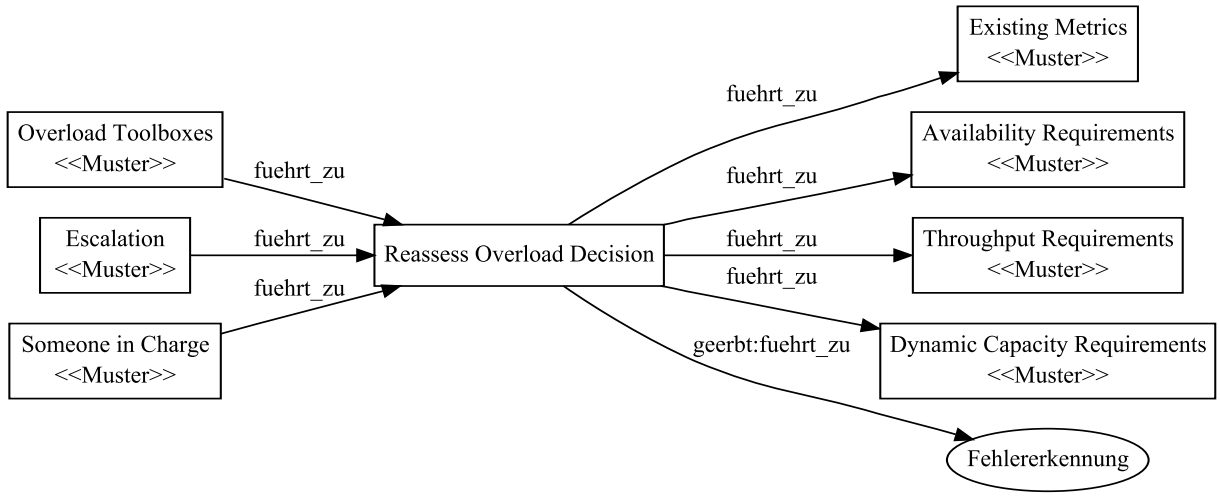
Gruppe(n)

- Fehlerkompensation
- Ueberarbeiten der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Reassess Overload Decision" befasst sich mit der Frage, was zu tun ist, wenn ein System mit den bestehenden Überlast-Kompensationsmechanismen nicht mehr in der Lage ist, die auftretende Systemlast zu tragen. Dazu schlägt es vor, eine Rückkopplung einzubauen, die dem System ggf. eine automatische Anpassung durch die Wahl einer anderen Kombination von Mechanismen ermöglicht.

Beziehungen - Grafik




Beziehungen - Detail

- **Overload Toolboxes -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Escalation -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Someone in Charge -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Reassess Overload Decision -> Existing Metrics** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Reassess Overload Decision -> Availability Requirements** : Aktualisieren der Anforderungen
  Das Reassess-Overload-Decision-Pattern behandelt die Anpassung der Auswahl von Mitteln zur Behandlung von durch Überlastung des Systems eingetretenen Problemen. Die Mittel zur Behandlung von Überlast-Situationen werden auf Basis der Verfügbarkeitsanforderungen (Availability Requirements) ausgewählt und umgesetzt. Deshalb müssen die Verfügbarkeitsanforderungen im Fall veränderter Overload Decisions angepasst werden.
- **Reassess Overload Decision -> Throughput Requirements** : Anpassen der Anforderungen
  Das Reassess-Overload-Decision-Pattern behandelt die Anpassung der Auswahl von Mitteln zur Behandlung von durch Überlastung des Systems eingetretenen Problemen. Das Eintreffen einer großen Zahl von Anfragen, Nachrichten oder allgemein Daten ist ein typischer Grund für hohe Systembelastung. Die Menge an Anfragen, Nachrichten oder Daten, die ein System zu einem Zeitpunkt bewältigt, wird als Durchsatz bezeichnet. Verändert sich der Umfang zu erwartender Überlastszenarien so sind folglich die Anforderungen an den Durchsatz (Throughput Requirements) anzupassen.

- **Reassess Overload Decision -> Dynamic Capacity Requirements** : anpassen
  Das Reassess-Overload-Decision-Pattern beschreibt die situationsbedingte Nachbesserung der Mechanismen zur Bewältigung von Überlastungsszenarien. Wird im Rahmen dieses Reassessment-Vorgangs eine dauerhaft höhere Systemlast festgestellt, so führt diese Neubewertung zu einer Anpassung der Dynamic Capacity Requirements.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *154. Recovery Blocks*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 53-56

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

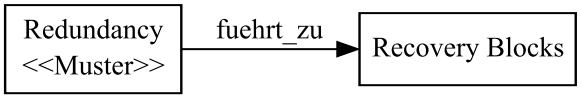
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Recovery Blocks" beschreibt das Verfahren, alternative Implementierungen für kritische Code-Blöcke vorzuhalten, um im Fehlerfall zu versuchen mit diesen alternativen Implementierungen doch noch ein valides Ergebnis zu ermitteln. Dazu wird die Reihenfolge der Verwendung der Alternativen festgelegt und bei der ersten Implementierung begonnen. Die jeweilige Folge-Implementierung wird nur dann genutzt, wenn vorher ein Fehler aufgetreten ist.

Beziehungen - Grafik




Beziehungen - Detail

- **Redundancy -> Recovery Blocks** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35

## *155. Redundancy*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 47-52

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

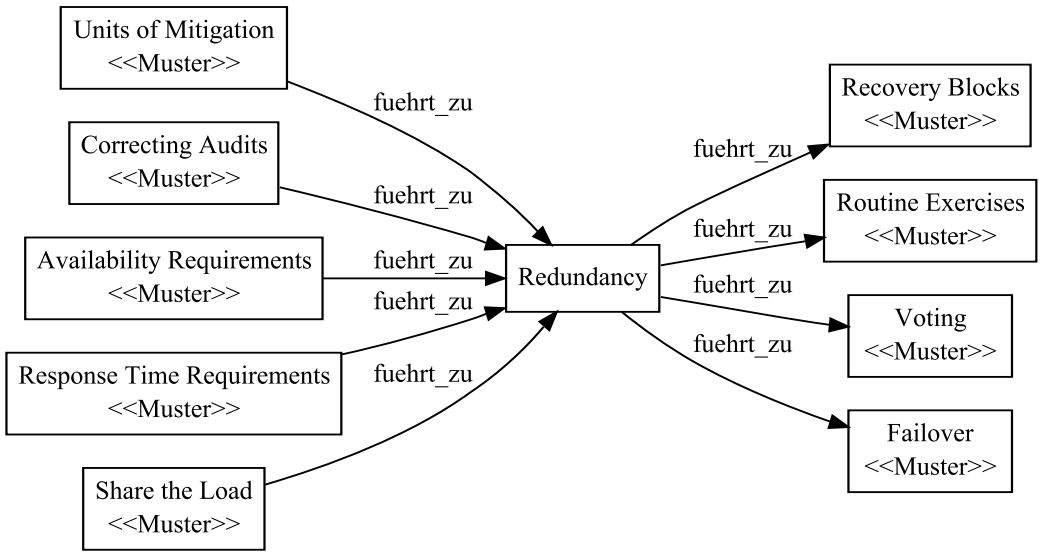
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Redundancy" beschreibt den Ansatz, Dinge (Server, Komponenten, Routinen) mehrfach vorzuhalten um zwischen ihnen die Last zu verteilen, die Ausfallsicherheit zu erhöhen oder die Richtigkeit von Ergebnissen zu prüfen.

Beziehungen - Grafik




Beziehungen - Detail

- **Units of Mitigation -> Redundancy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Correcting Audits -> Redundancy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Availability Requirements -> Redundancy** : umsetzbar durch
  Verfügbarkeitsanforderungen können unter anderem durch die Verwendung redundanter Komponenten erreicht werden. Sollten die Verfügbarkeitsanforderungen

durch reduzierte Update-Dauer, Online-Backups usw. nicht erreicht werden können, so können durch eine redundante Auslegung des Systems einige der Aktionen, die ohne redundante Auslegung zu einer Downtime führen würden, im Hintergrund ausgeführt werden.
Siehe S. 231, Software Requirement Patterns, Withall; Punkt: Replicate Hardware und S. 230, Software Requirement Patterns, Withall; Punkt: Machine shutdown without interrupting system; (nur durch Redundanz erreichbar)

- **Response Time Requirements -> Redundancy** : umsetzbar durch
  Anforderungen an die Antwortzeit, die bei einer vorgegebenen Lastsituation nicht durch eine einzelne Installation erfüllt werden können, lassen sich unter Umständen durch die Verteilung der Last auf eine oder mehrere redundante Installationen erfüllen.
- **Share the Load -> Redundancy** : Erreichung von Gleichen, die sich die Last teilen können
  Um das Share-the-Load-Pattern anzugwenden ist es notwendig, dass redundante Systembestandteile existieren, zwischen denen die Last verteilt werden kann.
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 206
- **Redundancy -> Recovery Blocks** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Redundancy -> Routine Exercises** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Redundancy -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Redundancy -> Failover** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *156. Registry*

Quellen

**Patterns of Enterprise Application Architecture, Fowler**

S. 480-485

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

Schwerpunkt(e)

- Flexibilität

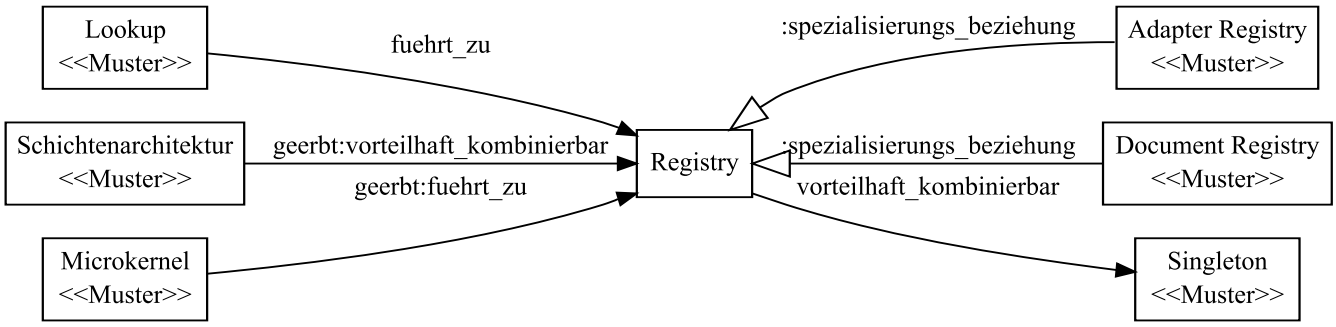
Gruppe(n)

- Kopplung und Entkopplung

## Kurzbeschreibung

Das Pattern "Registry" beschreibt den Ansatz eine Klasse zur Verfügung zu stellen, mittels deren Objekte andere, abhängige Objekte gesucht und gefunden werden können.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Lookup -> Registry** : Registry als Datenquelle
  Eine Registry ist eine mögliche Datenquelle für die Durchführung eines Lookup.
  Beispiel: RMIRegistry in Java
  (http://download.oracle.com/javase/6/docs/api/java/rmi/registry/Registry.html)
- **Registry -> Adapter Registry** : Idee: Registrierung von Objekten
  Das Registry-Pattern wird zur Verwaltung der Adapter-Objekte in der Adapter Registry verwendet.
- **Registry -> Document Registry** : Ist eine Spezialform von
- **Registry -> Singleton** : Häufig kombiniert
  Quelle: Patterns of Enterprise Application Architecture, Fowler, S. 481 (3. Absatz)
  *"For a process-scoped Registry, then, the usual option is a singleton."*
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Microkernel -> Kopplung und Entkopplung** : Entkopplung von Kern und Erweiterungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Eine Microkernel-Architektur ist mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns werden verwendet um die Erweiterungen und den Kern voneinander zu entkoppeln.

## *157. Regular Expression based Validation*

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

- Eingabevalidierung

Kontext
Bei der Eingabe von Daten sollen durch eine sofortige Prüfung eingegebener Werte Fehler verhindert werden. Dabei gibt es unterschiedliche Arten von Werten, die geprüft werden können. Einerseits sind Werte verfügbar bei denen durch bloße Anwendung einer Regel auf den Wert, ohne Zuhilfenahme dritter Daten, deren Gültigkeit festgestellt werden kann. Andererseits gibt es auch Werte, bei denen die Gültigkeit durch den Abgleich mit einer Quellmenge überprüft werden kann. In diesem Fall beschreibt die Regel die Ermittlung und den Vergleich mit der Quellmenge.

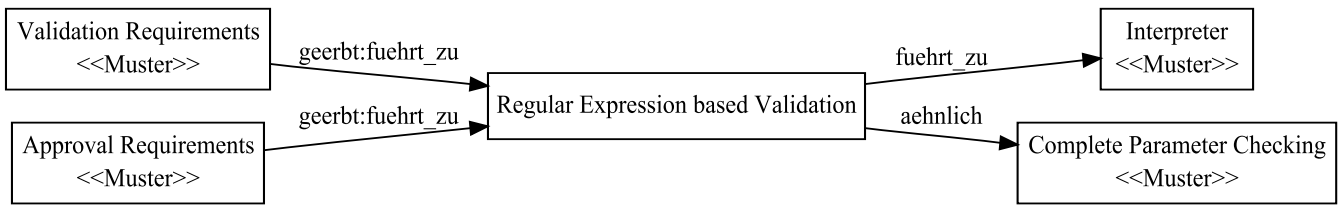
Problem
Die Gültigkeit eines Wertes kann allein durch die Anwendung einer Regel und den Wert selbst ermittelt werden. Der vorliegende Wert ist eine Zeichenkette. Sie muss einer bestimmten Form genügen, um gültig zu sein.

Lösung
Die Regel zur Validierung kann als Ausdruck formuliert werden, der den Wert eindeutig als Element einer Klasse von Zeichenketten identifizierbar macht. Dazu sind in den verschiedenen gängigen Programmiersprachen Funktionen zur Anwendung von regulären Ausdrücken enthalten. Reguläre Ausdrücke können verwendet werden, um abstrakte Muster zu definieren, die jeweils eine Klasse von Zeichenketten beschreiben. Wird ein regulärer Ausdruck auf eine konkrete Zeichenkette angewendet, so lässt sich eindeutig bestimmen, ob die Zeichenkette dem beschriebenen Muster entspricht. Dadurch können reguläre Ausdrücke im Rahmen der Validierung eingegebener Werte als Sprache zur Beschreibung der Gültigkeitsregel verwendet werden.

Beispiele (vgl. diverse Validator-Komponenten in Web-Frameworks)

- Die eingegebene eMail-Adresse hat ein gültiges Format.
- Die angegebene Telefonnummer ist gültig formatiert.
- geiches für Kontonummern, Postleitzahlen, Kostenstellen, usw.

Beziehungen - Grafik



Beziehungen - Detail

- **Kandidat: Regular Expression based Validation -> Interpreter** : Interpretation der Regel.
  Um einen regulären Ausdruck auf eine Zeichenkette anzuwenden, wird ein Interpreter benötigt, der den regulären Ausdruck einliest und somit auf den String anwendbar macht.
- **Validation Requirements -> Eingabevalidierung** : Bereich: Dateneingabe
  Geerbte Beziehung: fuehrt_zu_beziehung
  Ein Bereich der Validation Requirements befasst sich mit der Validierung von Daten während deren Eingabe bzw. Übertragung an das System, in dem die Validierung durchgeführt wird.
- **Approval Requirements -> Eingabevalidierung** : Anforderungen für spezielle Eingabevalidierung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Nutzung eines Approval Workflow, also der Bestätigung von eingegebenen Werten durch einen weiteren Benutzer (i.d.R. einen fachlichen Experten), ist eine spezielle Form der Eingabevalidierung.

## *158. Reintegration*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 233-235

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

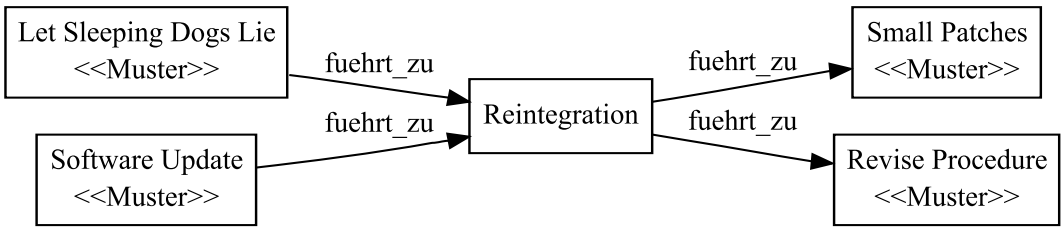
Gruppe(n)

- Fehlerkorrektur

Kurzbeschreibung

Das Pattern "Reintegration" befasst sich mit der Frage, wie korrigierte Versionen von Komponenten oder sonstige korrigierte Code-Bestandteile nach der erfolgten Korrektur wieder in das System integriert werden sollen.

Beziehungen - Grafik




Beziehungen - Detail

- **Let Sleeping Dogs Lie -> Reintegration** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Software Update -> Reintegration** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Reintegration -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Reintegration -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

## *159. Remote Procedure Invocation*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 50-52

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

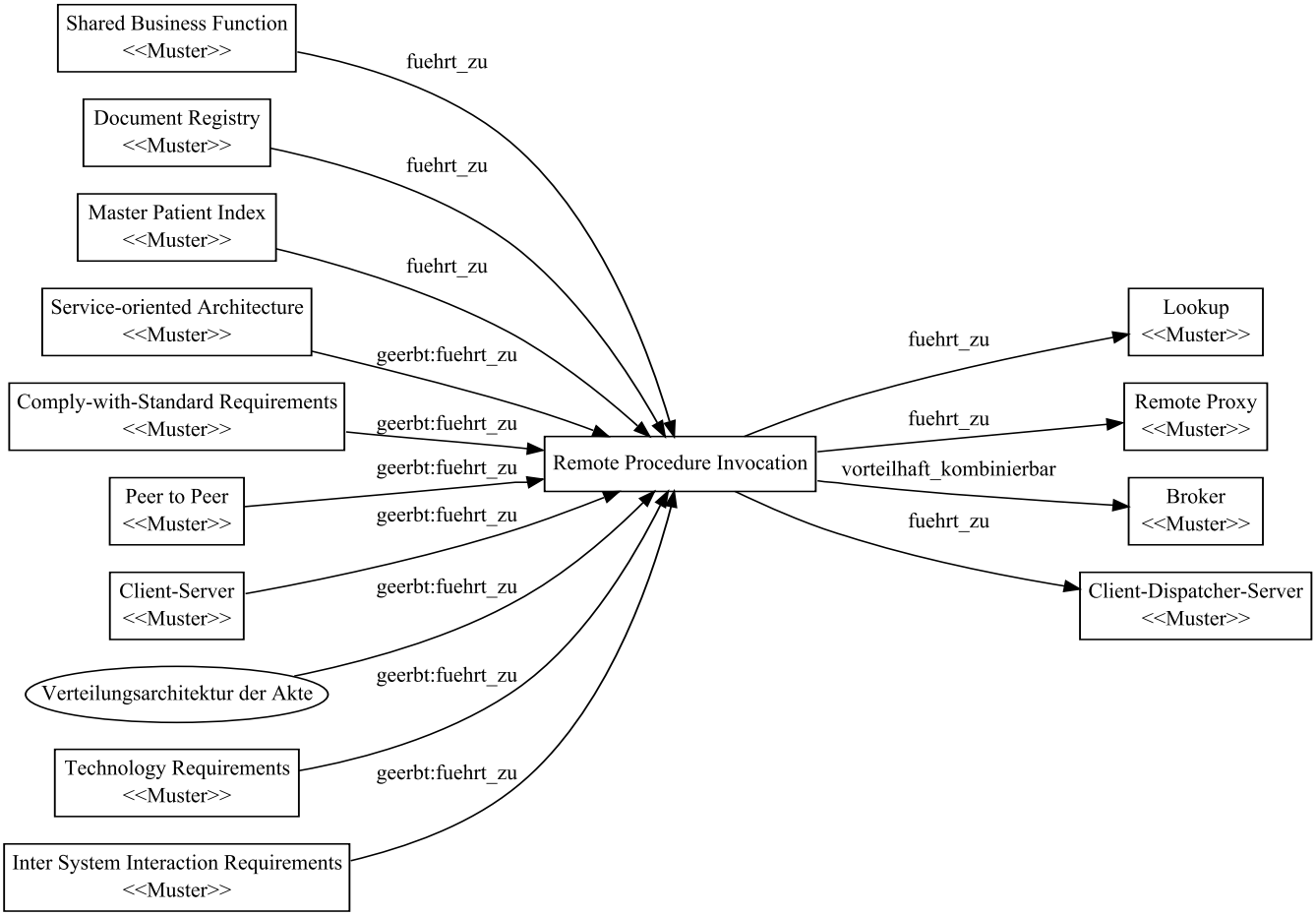
Gruppe(n)

- Synchrone Kommunikation

Kurzbeschreibung

Das Pattern "Remote Procedure Invocation" beschreibt den Ansatz, auf mehrere Rechner (oder zumindest mehrere Prozesse) verteilte Bestandteile von Softwaresystemen mittels über das Netz aufrufbare Funktionen zu verknüpfen.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Shared Business Function -> Remote Procedure Invocation** : umsetzbar durch
  Erschließt sich aus dem Text von S. 7 in Enterprise Integration Patterns; Hohpe.
- **Document Registry -> Remote Procedure Invocation** : Aufruf der Registry Methoden
  Um die verschiedenen Methoden der Document Registry von verschiedenen Systemknoten aus aufrufen zu können ist es notwendig, dass diese Methoden remotefähig zur Verfügung stehen.
- **Master Patient Index -> Remote Procedure Invocation** : Notwendig
  Um einen Master Patient Index umzusetzen ist es notwendig, dass dessen Schnittstelle von entfernten Softwaresystemen aufgerufen werden kann. Die entfernten Aufrufe dienen sowohl der Abfrage von IDs eines Patienten in anderen Systemen als auch der Registrierung von IDs für neue Patienten.
- **Remote Procedure Invocation -> Lookup** : Bestandteil wenn symbolische Namen verwendet werden
  Wenn symbolische Namen verwendet werden ist Lookup ein Bestandteil der Remote Procedure Invocation.
- **Remote Procedure Invocation -> Remote Proxy** : zwingender Bestandteil
  Ein Proxy ist ein zwingender Bestandteil, der Remote Procedure Invocation erst

komfortabel möglich macht. Beispiele hierfür sind nahezu alle gängigen RPC oder RMI APIs wie z. B. Java-RMI, CORBA, Webservices mit SOAP oder REST usw.

- **Remote Procedure Invocation -> Broker** : Entkopplung
  Zur Entkopplung von Server und Client kann eine Broker-Komponente bzw. Broker-Schicht eingeführt werden, die zwischen den beiden vermittelt. Siehe S. 99 in Pattern Oriented Software Architecture, Volume 1
- **Remote Procedure Invocation -> Client-Dispatcher-Server** : Ortstransparenz
  Die Ortstransparenz ist die zentrale Zielsetzung von Client-Dispatcher-Server. Dieser Zusammenhang wird in Pattern Oriented Software Architecture, Volume 1, S. 324, Abschnitt Solution beschrieben.
- **Service-oriented Architecture -> Synchrone Kommunikation** : für synchrone Dienste
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: SOA in Practice, Josuttis; S. 124; Message Exchange Pattern: Request/Response;
  Das in der Quelle dargestellte Message-Exchange-Pattern beschreibt einen synchronen RPC-ähnlichen Aufruf durch Request/Reply Kommunikation.
- **Comply-with-Standard Requirements -> Synchrone Kommunikation** : wenn im Standard gefordert
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Peer to Peer -> Synchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Peer-to-Peer-Kommunikation ist mit mitteln synchroner Kommunikation realisierbar.
- **Client-Server -> Synchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Client-Server-Systeme sind mit Mitteln synchroner Kommunikation realisierbar
- **Verteilungsarchitektur der Akte -> Synchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.
- **Technology Requirements -> Synchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Inter System Interaction Requirements -> Synchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung

Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.

## *160. Remote Proxy*

Quellen

**POSA 4, A Pattern Language for Distributed Computing, Buschmann**

S. 240 ff. als Client Proxy

Pattern-Oriented Software Architecture, 4, A Pattern Language for Distributed Computing, Frank Buschmann, Kevlin Henney, Douglas C. Schmidt, Wiley Series in Software Design Patterns, 2007

**POSA 1, A System of Patterns, Buschmann**

S. 268 erwähnt als Ausprägung des Proxy-Pattern

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

Schwerpunkt(e)

- Kommunikation
- Flexibilität

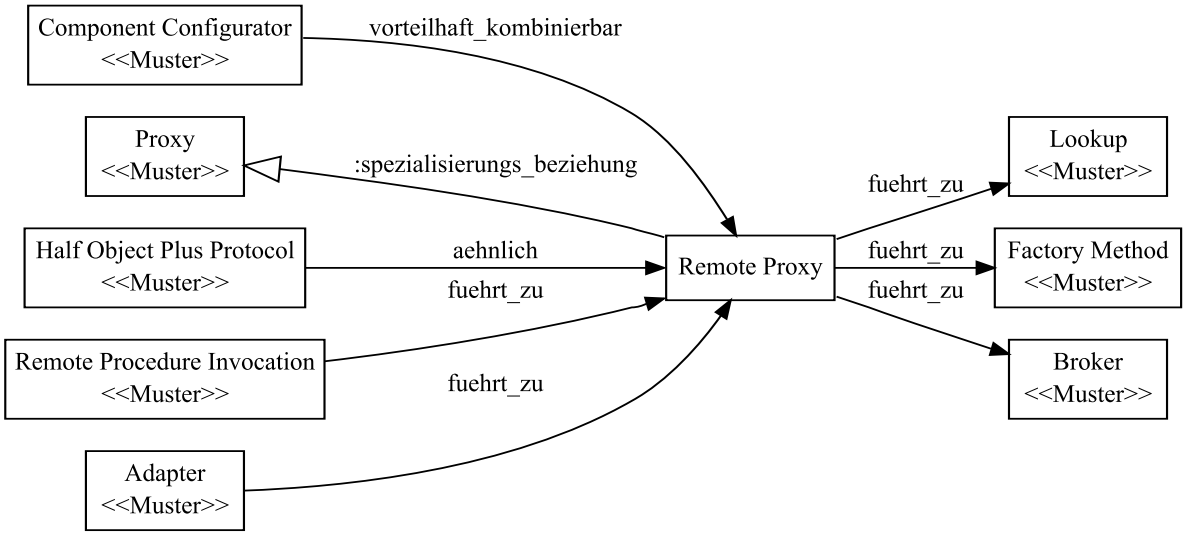
Gruppe(n)

- Handle Body Patterns
- Verbergen der Kommunikation

Kurzbeschreibung

Das Pattern "Remote Proxy" beschreibt den Ansatz, die Kommunikation über das Netz, die bei Aufruf einer entfernten Funktion notwendig ist, hinter einem lokalen, der aufgerufenen Komponente (oder dem aufgerufenen Objekt) bezüglich seiner Schnittstelle gleichen Stellvertreter zu verbergen.

Beziehungen - Grafik




Beziehungen - Detail

- **Component Configurator -> Remote Proxy** : Anbindung entfernter Komponenten
  Zur konfigurierbaren Anbindung entfernter Komponenten (z. B. von Webservices etc.) kann das Component-Configurator-Pattern vorteilhaft mit dem Remote-Proxy-Pattern kombiniert werden.
- **Proxy -> Remote Proxy** :
  Eine Spezialform des Proxy ist der Remote Proxy. Er dient der Kapselung entfernter Kommunikation in einem lokalen, dem entfernten Objekt bezüglich seiner Schnittstelle äquivalenten Stellvertreter-Objekt.
- **Half Object Plus Protocol -> Remote Proxy** : Lösen das gleiche Problem, komfortable entfernte Kommunikation
  Quelle: Portland Pattern Repository; http://c2.com/cgi/wiki?HalfObjectPlusProtocol
  Letztmalig geprüft: 13.10.2011
  Letzte Änderung laut Seite: 09.07.2003
  In Beschreibung zu Half Object Plus Protocol: „*Patterns which solve the same problem include RemoteProxy (which delegates all requests back to a single site)*“
- **Remote Procedure Invocation -> Remote Proxy** : zwingender Bestandteil
  Ein Proxy ist ein zwingender Bestandteil, der Remote Procedure Invocation erst komfortabel möglich macht. Beispiele hierfür sind nahezu alle gängigen RPC oder RMI APIs wie z. B. Java-RMI, CORBA, Webservices mit SOAP oder REST usw.
- **Adapter -> Remote Proxy** : Wenn Adaptee Remote
  Wenn das anzupassende Objekt (Adaptee) sich auf einem entfernten System befindet, ist es notwendig, die Adapter-Implementierung mit dem Remote-Proxy-Pattern oder dem Broker-Pattern zu kombinieren.
- **Remote Proxy -> Lookup** : Instanzierung
  Remote Proxy benötigt Lookup oder Factory zur Instanzierung wenn nicht bei jeder Instanzierung vollständige Verbindungsdaten zur entfernten Implementierung als Parameter übergeben werden sollen.

- **Remote Proxy -> Factory Method** : Instanzierung
  Remote Proxy benötigt Lookup oder Factory zur Instanzierung wenn nicht bei jeder Instanzierung vollständige Verbindungsdaten zur entfernten Implementierung als Parameter übergeben werden sollen.
- **Remote Proxy -> Broker** : Zur Vereinfachung der Proxy-Methoden
  Pattern Oriented Software Architecture, Volume 4 beschreibt auf S. 237, untere Abbildung, die Verwendung von Brokern auf Client- und Serverseite zur Vereinfachung der Kommunikation. Dabei wird der Code für die eigentliche Kommunikation aus den einzelnen Proxy-Methoden in den Broker der Clientseite verlagert.

## *161. Remote Storage*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 171-173

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

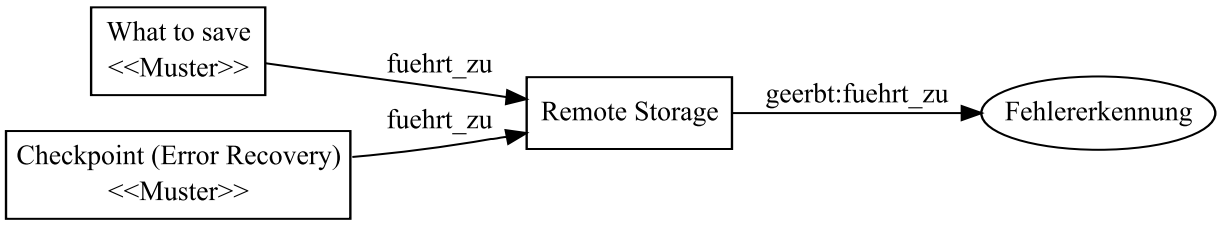
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "Remote Storage" setzt sich mit der Frage auseinander, wo in redundant ausgelegten Systemen Checkpoints (Error Recovery) gespeichert werden sollen, und zwar außerhalb des Rechners, von dem sie geschrieben werden, um im Falle eines Ausfalls das redundante Pedant schnell auf den entsprechenden Zustand setzen zu können.

Beziehungen - Grafik




Beziehungen - Detail

- **What to save -> Remote Storage** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Checkpoint (Error recovery) -> Remote Storage** : Möglicher Speicherort
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 171
  *"What storage location should be used for Checkpoints to reduce the time before execution can be resumed after error recovery?"*
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *162. Reproducible Error*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 236-237

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

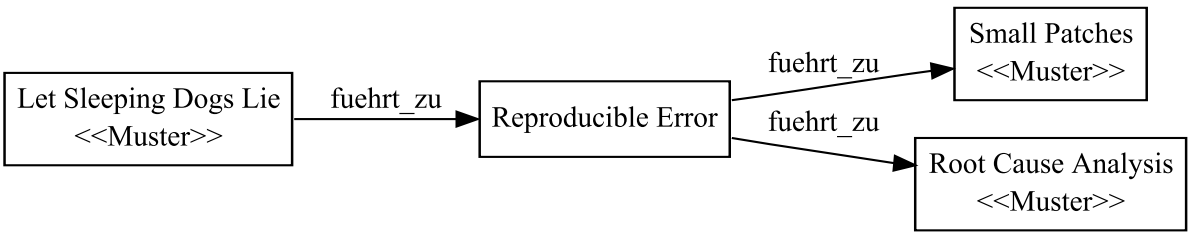
Gruppe(n)

- Fehleranalyse

Kurzbeschreibung

Das Pattern "Reproducible Error" setzt sich mit der Reproduzierbarkeit von Fehlern im Rahmen der Fehleranalyse auseinander.

Beziehungen - Grafik




Beziehungen - Detail

- **Let Sleeping Dogs Lie -> Reproducible Error** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Reproducible Error -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

- **Reproducible Error -> Root Cause Analysis** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

## *163. Response Time Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 195-204

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation
- Zuverlässigkeit

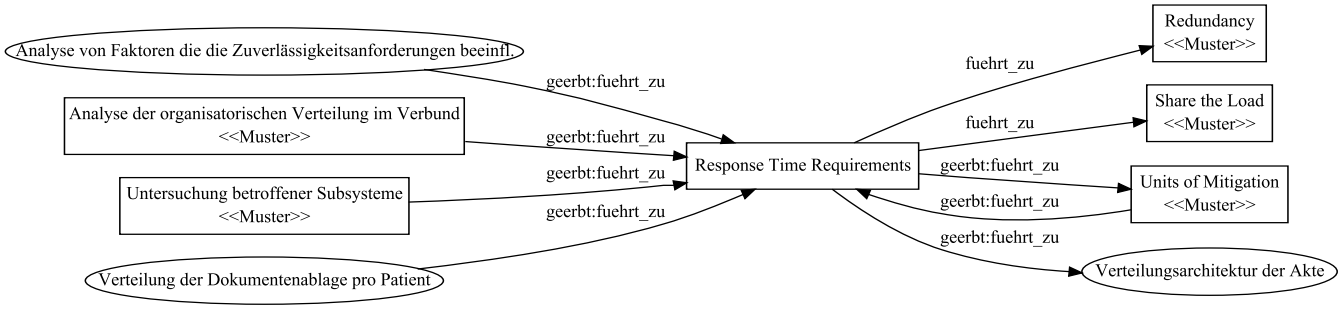
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen
- Kommunikationsanforderungen

Kurzbeschreibung

Das Pattern "Response Time Requirements" erläutert die Definition von Anforderungen an das Antwortzeit-Verhalten, das innerhalb einer Anwendung unter bestimmten Rahmenbedingungen erfüllt werden muss.

Beziehungen - Grafik




Beziehungen - Detail

- **Response Time Requirements -> Redundancy** : umsetzbar durch
  Anforderungen an die Antwortzeit, die bei einer vorgegebenen Lastsituation nicht durch eine einzelne Installation erfüllt werden können, lassen sich unter Umständen durch die Verteilung der Last auf eine oder mehrere redundante Installationen erfüllen.
- **Response Time Requirements -> Share the Load** : umsetzbar durch
  Um auch bei Lasten, die die Leistungsfähigkeit eines einzelnen Knotens überschreiten, die Response Time Requirements einzuhalten, kann die Last zwischen redundanten

Knoten verteilt werden. Diese Lösung setzt eine gleichzeitige Verwendung des Redundancy Patterns voraus.

- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *164. Restart*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 151-153

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

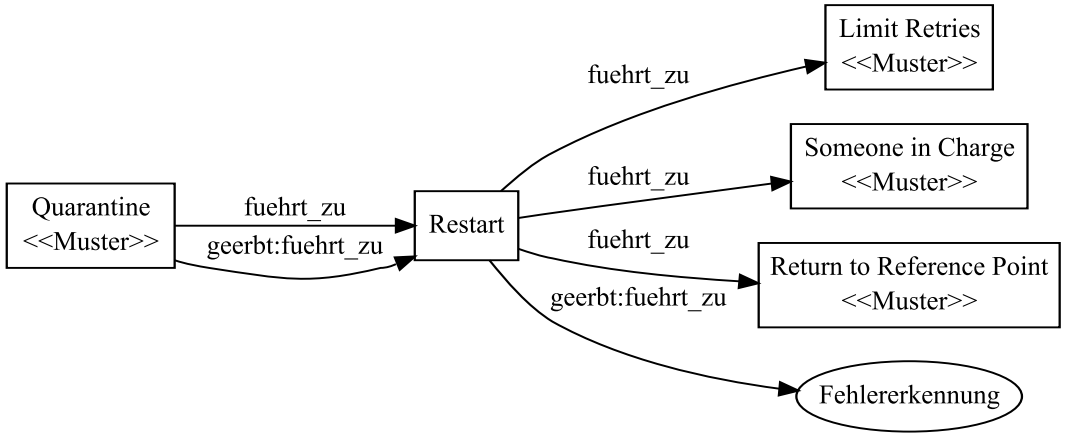
Gruppe(n)

- Automatisierte Fehlerbehebung
- Recovery Types

Kurzbeschreibung

Das Pattern "Restart" beschreibt den Fall, in dem eine klassische Behebung oder Kompensation des Fehlers gescheitert ist. Abhängig von der Art des Fehlers usw. lässt sich der Betrieb evtl. durch ein neu Starten des Systems oder neu Starten der Routine, einschließlich des damit einhergehenden Datenverlusts, wieder herstellen.

Beziehungen - Grafik




Beziehungen - Detail

- **Quarantine -> Restart** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
  **Hinweis:** Restart ist eine Lösung, die nur zur Kompensation von Fehlern bei

transaktionalen, idempotenten und zustandsinvarianten Diensten verwendet werden kann. Bei Diensten, Methoden oder Funktionen, die diese Eigenschaften nicht erfüllen kann die wiederholte Ausführung zur Produktion fehlerhafter Daten führen.

- **Restart -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Restart -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Restart -> Return to Reference Point** : Bei zustandsverändernden Operationen
  Soll eine zustandsverändernde Operation durch Restart nach einem Fehler nochmals aufgerufen werden, so ist es notwendig den Zustand des Systems auf einen Referenzpunkt vor der Ausführung der Operation zurückzusetzen, da ansonsten evtl. Änderungen doppelt ausgeführt werden.
- **Quarantine -> Recovery Types** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *165. Return to Reference Point*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 158-159

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

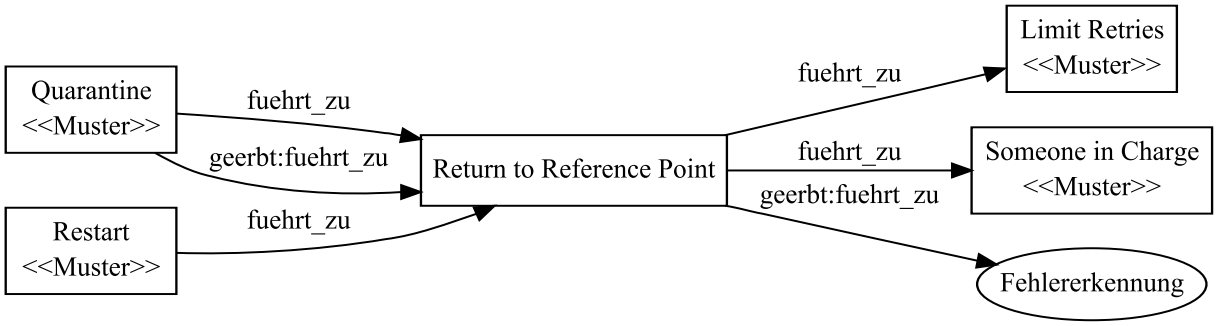
Gruppe(n)

- Automatisierte Fehlerbehebung
- Recovery Types

Kurzbeschreibung

Das Pattern "Return to Reference Point" ist von seinen Auswirkungen zwischen Rollback bzw. Roll-Forward und Restart anzusiedeln. Es beschreibt das zurücksetzen des Systems auf einen definierten früheren Referenz-Checkpoint.

Beziehungen - Grafik




Beziehungen - Detail

- **Quarantine -> Return to Reference Point** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Restart -> Return to Reference Point** : Bei zustandsverändernden Operationen
  Soll eine zustandsverändernde Operation durch Restart nach einem Fehler nochmals aufgerufen werden, so ist es notwendig den Zustand des Systems auf einen Referenzpunkt vor der Ausführung der Operation zurückzusetzen, da ansonsten evtl. Änderungen doppelt ausgeführt werden.
- **Return to Reference Point -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Return to Reference Point -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Quarantine -> Recovery Types** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *166. Reverse Proxy*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 457-464; Spezialform Protection Reverse Proxy

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 465-472; Spezialform Integration Reverse Proxy

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

### Schwerpunkt(e)

- Sicherheit

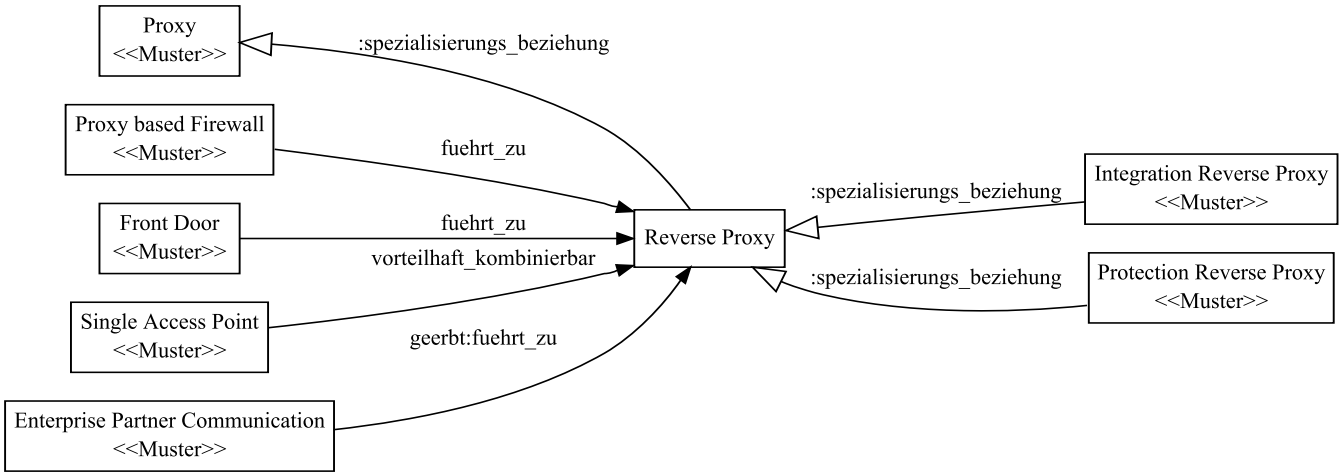
### Gruppe(n)

- Access Restriction

### Kurzbeschreibung

Das Pattern "Reverse Proxy" subsumiert die verschiedenen verfügbaren Reverse-Proxy-Patterns wie z. B. Integration Reverse Proxy und Protection Reverse Proxy, denn alle Reverse-Proxy-Patterns sind technisch sehr ähnlich und unterscheiden sich vorrangig in der damit verbundenen Aufgabe oder Zielsetzung.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Proxy -> Reverse Proxy** :
  Eine Spezialform des Proxy ist der Reverse Proxy. Beim Reverse Proxy handelt es sich üblicherweise nicht um einen Stellvertreter für ein Objekt, sondern um einen Stellvertreter für einen oder mehrere Server. Der Reverse Proxy macht deren Inhalte in einem Netz verfügbar, aus dem man diese Server nicht direkt erreichen kann.

- **Proxy based Firewall -> Reverse Proxy** : Im verteilten Szenario
  Zur Umsetzung einer Proxy based Firewall im verteilten Szenario ist ein Reverse Proxy empfehlenswert.
  Quelle: Security Patterns; Schumacher et al.; S.414
  *"Consider configurations such as Protection Reverse Proxy, Integration Reverse Proxy or a combination with a Packet Filter Firewal in a distributed configuration"*
- **Front Door -> Reverse Proxy** : Bestandteil
  Quelle: Security Patterns; Schumacher et al.; S. 473
  *"A reverse proxy is an ideal point to implement authentication and authorizadion, by implementing a Web entry server for your back-ends"*
  Quelle: Security Patterns; Schumacher et al.; S. 475
  *"Solution: Implement a Front Door server as a specialization of the Integration Reverse Proxy that identifies user and keeps track of user sessions.* [...] *,remember that it can also act as a Protection Reverse Proxy for the public part of the Web site."*
- **Single Access Point -> Reverse Proxy** :
  Ein Single Access Point kann durch eine Ausprägung von Reverse Proxy implementiert werden.
  Quelle: Security Patterns; Schumacher et al.; S. 286
  *"Other patterns in this book, such as the firewall patterns in Chapter 12 and Protection Reverse Proxy (457), provide examples of effective single access points, in which the clients are not always users, but can be network traffic that needs entry to the protected system."*
- **Reverse Proxy -> Integration Reverse Proxy** : Ist eine Spezialform von
  Der Integration Reverse Proxy ist ein spezieller Reverse Proxy, der als Stellvertreter für mehrere Systeme agiert.
- **Reverse Proxy -> Protection Reverse Proxy** : Ist eine Spezialform von
  Der Protection Reverse Proxy ist ein spezieller Reverse Proxy.
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung.
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *167. Revise Procedure*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S: 245-246

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

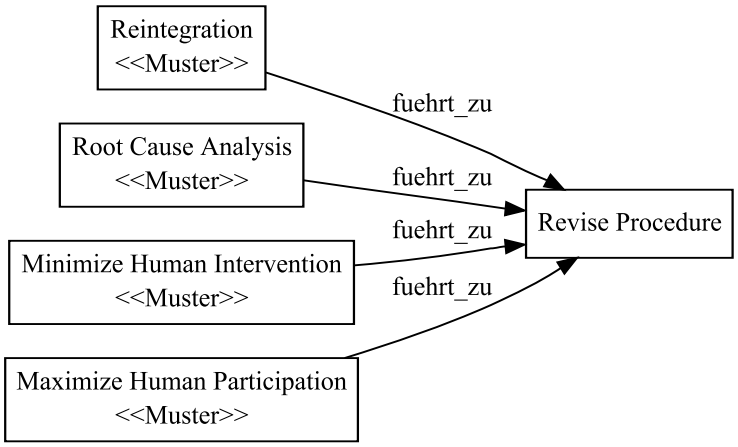
Gruppe(n)

- Fehleranalyse
- Ueberarbeiten der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Revise Procedure" befasst sich mit der Vermeidung von Fehlern, die durch Administratoren am System auftreten können.

Beziehungen - Grafik




Beziehungen - Detail

- **Reintegration -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Root Cause Analysis -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Minimize Human Intervention -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Maximize Human Participation -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *168. Riding over Transients*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 132-134

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

### Schwerpunkt(e)

- Zuverlässigkeit

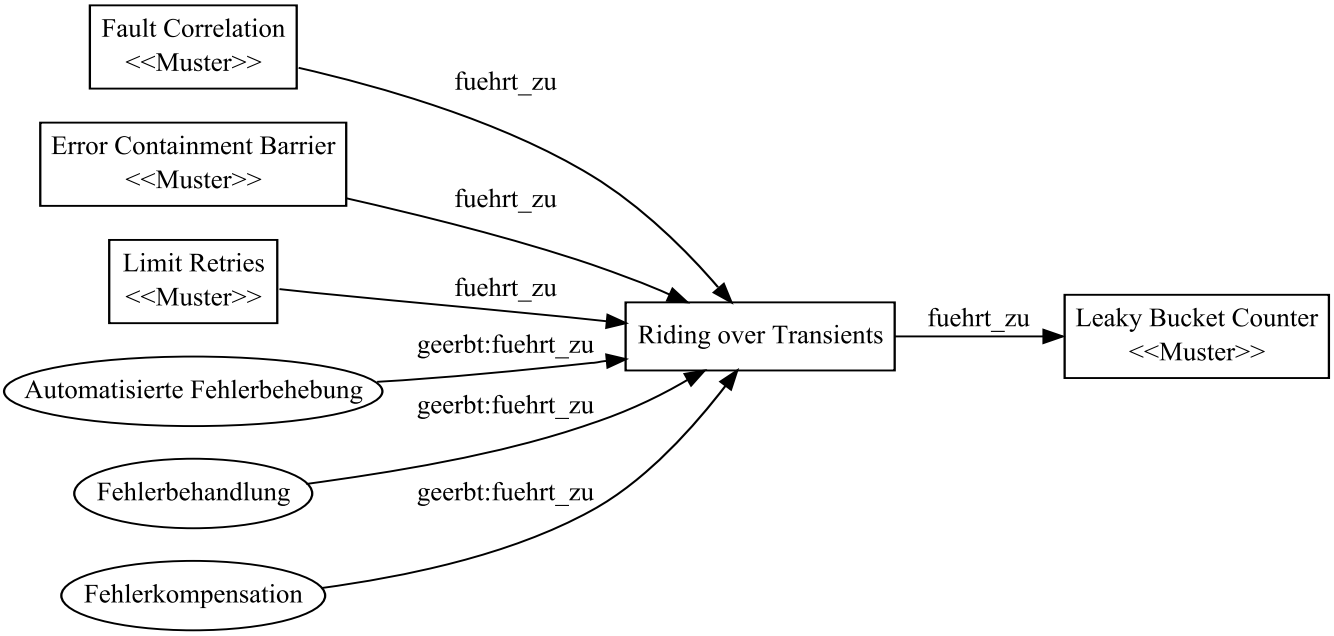
### Gruppe(n)

- Fehlererkennung

### Kurzbeschreibung

Das Pattern "Riding over Transients" befasst sich mit der Frage, wann Fehler, die keine längerfristigen Auswirkungen auf ein System haben ignoriert werden könne, um nicht unnötig Rechenleistung für deren Behandlung zu verbrauchen.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Fault Correlation -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Error Containment Barrier -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Limit Retries -> Riding over Transients** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Riding over Transients -> Leaky Bucket Counter** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu

auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *169. Risk Determination*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 137-147

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

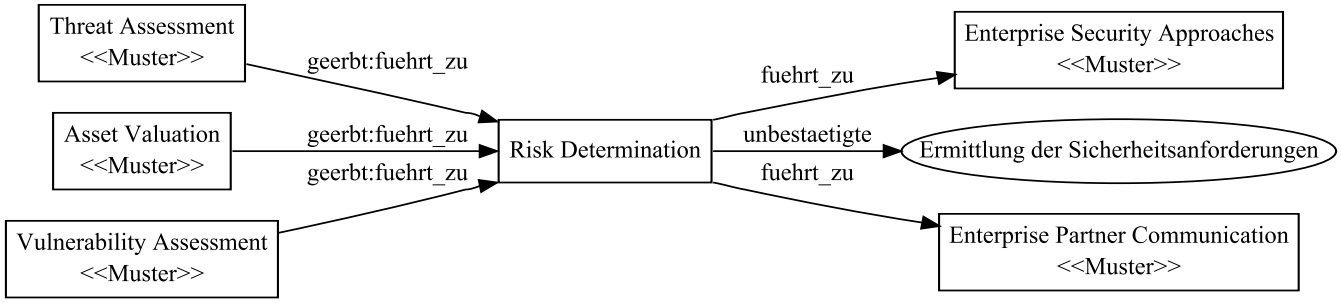
Gruppe(n)

- Bewertung der Risikosituation

Kurzbeschreibung

Das Pattern "Risk Determination" beschreibt einen Ansatz zur Risiko-Ermittlung und -Bewertung, dessen Ergebnis eine tabellarische Darstellung der bewerteten Risiken, die auf das zu planende System einwirken ist.

Beziehungen - Grafik



Beziehungen - Detail

- **Risk Determination -> Enterprise Security Approaches** : Input: Bewertete Risiken
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.
- **Risk Determination -> Enterprise Partner Communication** : Input: Bewertete Risiken
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Threat Assessment -> Bewertung der Risikosituation** : Input: Menge der identifizierten Bedrohungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns; Schumacher et al; S. 60 u. 61
  Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
  Quelle: Security Patterns; Schumacher et al; S. 147;Risk Determination, Liabilities: *"The results are based on the completeness and subjectivity of Asset Valuation, Threat Assessment and Vulnerability Assessment"*
- **Asset Valuation -> Bewertung der Risikosituation** : Input: bewertete bestehende Systeme und Inhalte
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 60 und 61 in Security Patterns, Schumacher et al.: Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
- **Vulnerability Assessment -> Bewertung der Risikosituation** : Input: Schwachstellen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 60 und 61 in Security Patterns, Schumacher et al.: Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
  Quelle: S. 147 in Security Patterns, Schumacher et al.;Risk Determination, Liabilities: "*The results are based on the completeness and subjectivity of Asset Valuation, Threat Assessment and Vulnerability Assessment"*

## *170. Role based Access Control*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 249-252

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

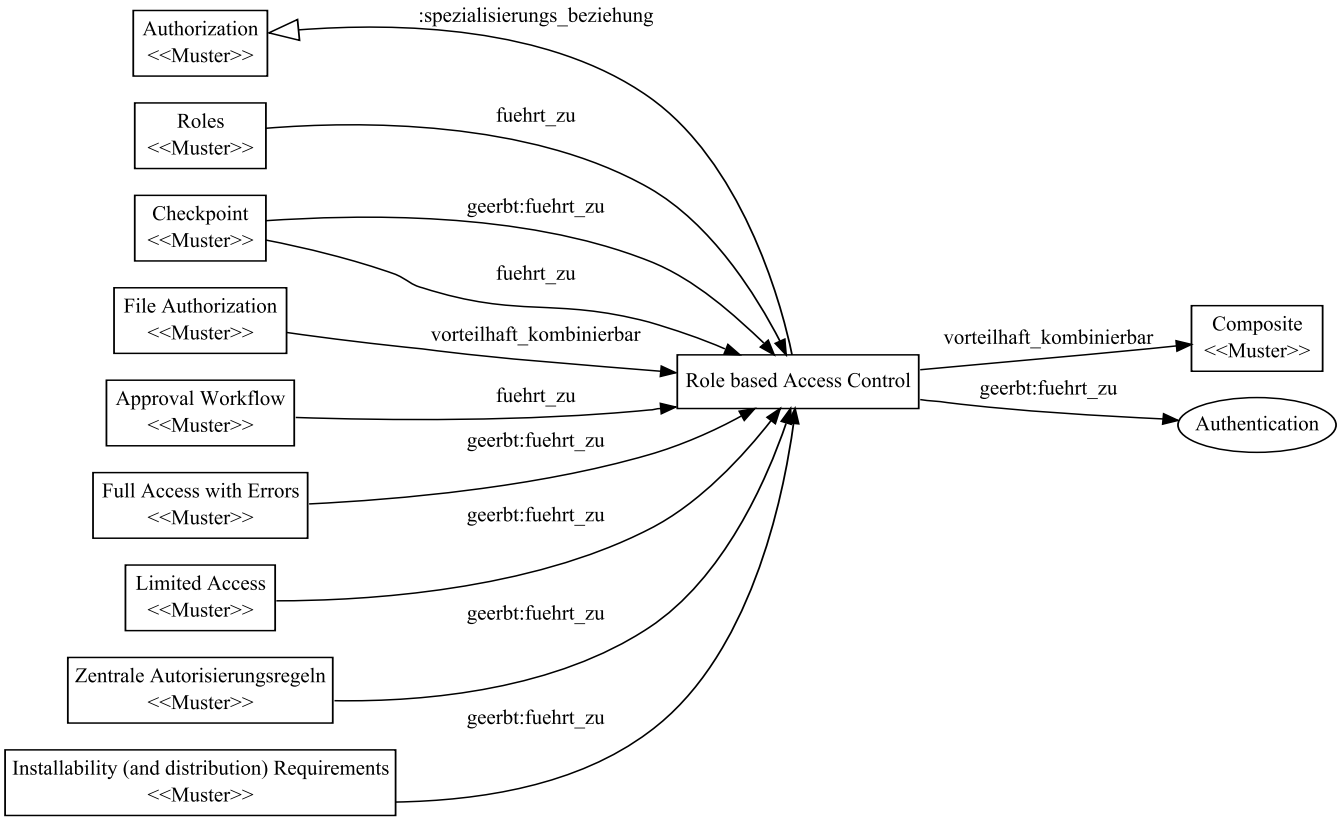
Gruppe(n)

- Access Control or Authorization

Kurzbeschreibung

Das Pattern "Role based Access Control" beschreibt den Ansatz, Subjekte die ein System nutzen zu aufgabenbezogenen Rollen zusammenzufassen und die Steuerung der Berechtigungen nicht individuell zwischen den Funktionen des Systems und den nutzenden Subjekten, sondern zwischen Funktionen und Rollen abzubilden.

Beziehungen - Grafik



Beziehungen - Detail

- **Authorization -> Role based Access Control** : Rechte durch Rollen
  Das Role-based-Access-Control-Pattern ist eines Spezialisierung des Authorization-Patterns. (Quelle: S. 248, Security Patterns, Schumacher et al.)
- **Roles -> Role based Access Control** : Berechtigungen technisch umsetzbar durch
  Quelle: u.A. Security Patterns; Schumacher et al.; S. 259
- **Checkpoint -> Role based Access Control** : Sicherheitsprüfungen
  Quelle: Security Patterns, Schumacher et al., S. 296;
  *"Role based Access Control is often used to implement Check Point's Security checks."*
- **File Authorization -> Role based Access Control** : Wenn Rollen genutzt werden
  Wenn für die Erteilung von Berechtigungen zum Dateizugriff Rollen verwendet werden sollen, ist das Role-based-Access-Control-Pattern notwendiger Bestandteil einer Lösung zur Umsetzung rollenbasierten Dateizugriffs.
  Quelle: Security Patterns; Schumacher et al.; S. 354
- **Approval Workflow -> Role based Access Control** : Rollenbasierte Zuordnung des Prüfenden
  Zuständigkeiten von Benutzern können im Zeitverlauf Änderungen unterworfen sein. Um zu vermeiden, dass aus diesen organisatorischen Änderungen auch Änderungen am Code von IT-Systemen folgen müssen, wird im Approval Workflow eine rollenbasierte Zuordnung von Akteuren angewendet.
- **Role based Access Control -> Composite** : Composit Roles
  Quelle: Security Patterns; Schumacher et al.; S. 251
  *"The model shown in the figure on page 252 additionally considers composite roles - it is an application of Composite - and separation of administration from other rights, an application of the policy of separation of duties."*
- **Full Access with Errors -> Access Control or Authorization** :
  Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung einer Zugriffsberechtigung setzt ein per Authentifizierung ermitteltes Subjekt (Benutzer oder Maschine) voraus. Um Full Access with Errors umzusetzen wird diese Berechtigungsinformation benötigt um im Bedarfsfall den Zugriff unter Hinweis auf einen Berechtigungsfehler unterbinden zu können.
- **Limited Access -> Access Control or Authorization** : Berechtigungsermittlung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung des Limited-Access-Patterns wird die Menge der für einen Benutzer, oder allgemeiner, für ein Subjekt verfügbaren Funktionen auf Basis von Berechtigungsinformation eingeschränkt. Dazu sind Mechanismen zur Ermittlung dieser Berechtigungsinformation notwendig.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Geerbte Beziehung: fuehrt_zu_beziehung
  Aktenbezogene Autorisierungskonzepte lassen sich durch Kombination von Patterns der Gruppe Access Control or Authorization umsetzen. Bei der Verwendung zentraler

Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.

- **Checkpoint -> Access Control or Authorization** : Notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al., S. 287;
  *"*[...]*, a means of identification and authentication and a response to unauthorized break-in attempts is required for securing the system."*
- **Installability (and distribution) Requirements -> Access Control or Authorization**
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 276, Software Requirement Patterns; Withall;
  Extra Requirements, Punkt 2. Authorization to install.
- **Access Control or Authorization -> Authentication** : benötigt zwingend
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um den Zugriff auf Objekte durch Autorisierungsregeln beschränken zu können ist zwingend die Authentifizierung und Authentisierung des zugreifenden Subjekts notwendig, da nur für identifizierbare Subjekte unterschiedliche Berechtigungen vergeben werden können.

## *171. Roles*

Quellen

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

S. 11

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 259-264 als Role Rights Definition

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

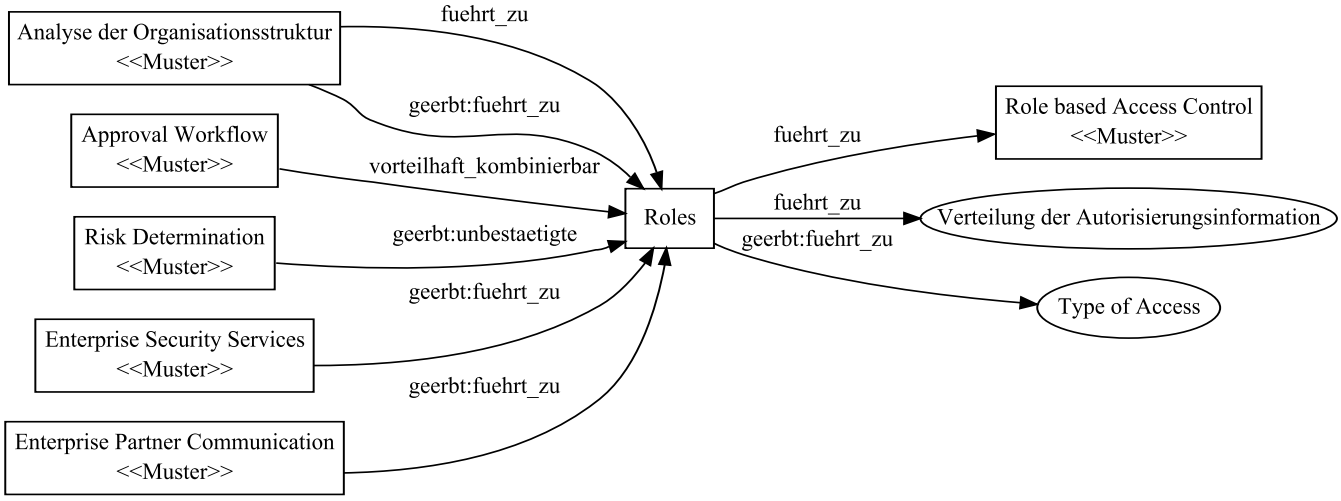
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "Roles" beschreibt das zusammenfassen der Nutzer eines Systems zu aufgabenbezogenen Gruppen, den Rollen.

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der Organisationsstruktur -> Roles** : Input: Akteure und Aufgabengebiete
  Das Analyse-der-Organisationsstruktur-Pattern liefert durch seine Anwendendung eine Auflistung von Akteuren und deren Aufgabengebiete. Aufgabe des Roles-Patterns ist es, diese zu abstrahieren um sie zu allgemeinen, für die Anforderungsdefinition standardisierten Rollen zusammenzufassen.
- **Approval Workflow -> Roles** : Vereinfachung des Bestätigungs- oder Freigabeworkflow
  Wird vor der Verwendung von Daten zu deren Validierung ein Freigabeworkflow eingesetzt, so kann die Definition der Freigabebedingungen durch die Verwendung von Rollen, die jeweils eine Menge von Benutzer repräsentieren, deutlich vereinfacht werden.
- **Roles -> Role based Access Control** : Berechtigungen technisch umsetzbar durch
  Quelle: u.A. Security Patterns; Schumacher et al.; S. 259
- **Roles -> Aktenbezogene Autorisierungskonzepte** : Organisationsbezug
  Das Roles-Pattern beschreibt die Ermittlung und Definition von Rollen und zugehörigen Berechtigungen für Benutzer eines Systems. Im komplexen Umfeld einer verteilten Krankenakte haben Benutzer diese Rollen aber nur bezogen auf bestimmte Organisationen oder Organisationseinheiten. Bei der Umsetzung aktenbezogener Autorisierungskonzepte mit zentral verwalteten Autorisierungsregeln ist deshalb auch die Berücksichtigung des Organisationsbezugs von Rollen und Rechten zu bedenken.
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.

- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** :
  Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *172. Roll-Forward*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 156-157

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

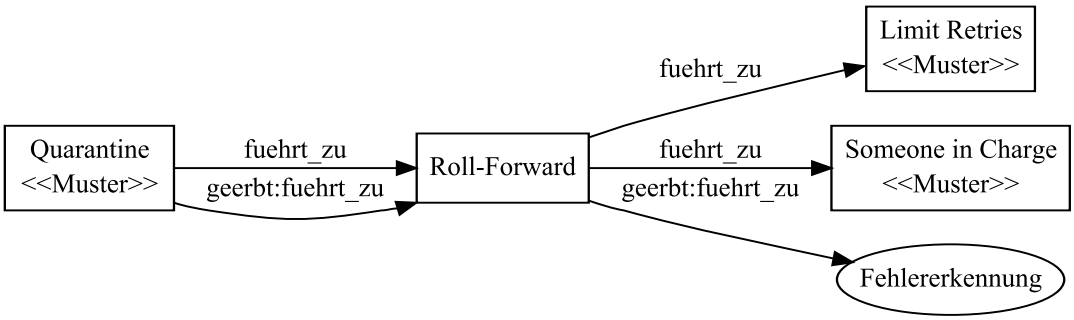
Gruppe(n)

- Automatisierte Fehlerbehebung
- Recovery Types

Kurzbeschreibung

Das Pattern "Roll-Forward" beschreibt den Ansatz, dass unter bestimmten Umständen ein Bereich einer Routine übersprungen werden kann, wenn in ihm Fehler aufgetreten sind. In diesem Fall wird die Ausführung an der nächsten Stelle, an der ohne das Ergebnis des fehlerhaften Bereichs weitergemacht werden kann fortgesetzt.

Beziehungen - Grafik




Beziehungen - Detail

- **Quarantine -> Roll-Forward** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Roll-Forward -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Roll-Forward -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Quarantine -> Recovery Types** :
  Geerbte Beziehung: fuehrt_zu_beziehung

Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *173. Rollback*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 154-155

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

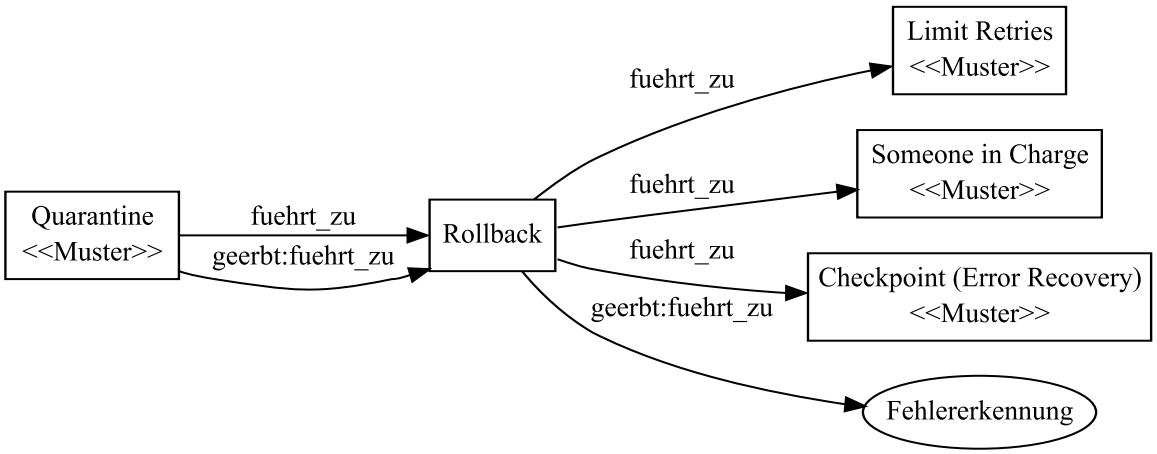
Gruppe(n)

- Automatisierte Fehlerbehebung
- Recovery Types

Kurzbeschreibung

Das Pattern "Rollback" beschreibt das Verfahren, den Zustand nach dem Auftreten eines Fehlers auf einen bekannt validen Zustand zurückzusetzen, sodass die Verarbeitung (i. d. R. anderer Anfragen) fortgesetzt werden kann.

Beziehungen - Grafik



Beziehungen - Detail

- **Quarantine -> Rollback** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141
- **Rollback -> Limit Retries** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 141; Abbildung 47
- **Rollback -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Rollback -> Checkpoint (Error recovery)** : Benötigt als Bestandteil
  Quelle: Patterns for Fault Tolerant Systems, Hanmer, S. 166;
  *"Many actions that might be taken to isolate or remediate an error will involve rolling the system back to a known place* [dieser bekannte Zustand kann durch die Anwendung des Checkpoint-Patterns erreicht werden] *through a Rollback* [...]*"*
- **Quarantine -> Recovery Types** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *174. Root Cause Analysis*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 242-244

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

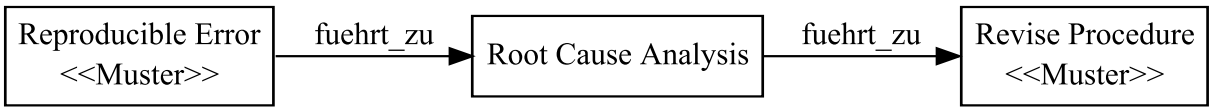
Gruppe(n)

- Fehleranalyse

Kurzbeschreibung

Das Pattern "Root Causes Analysis" beschreibt den iterativen Weg, dem Fehlerursprung über mehrere Stufen von Folgefehlern hinweg auf den Grund zu gehen.

Beziehungen - Grafik




Beziehungen - Detail

- **Reproducible Error -> Root Cause Analysis** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Root Cause Analysis -> Revise Procedure** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228

## *175. Routine Audits*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 127-128

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

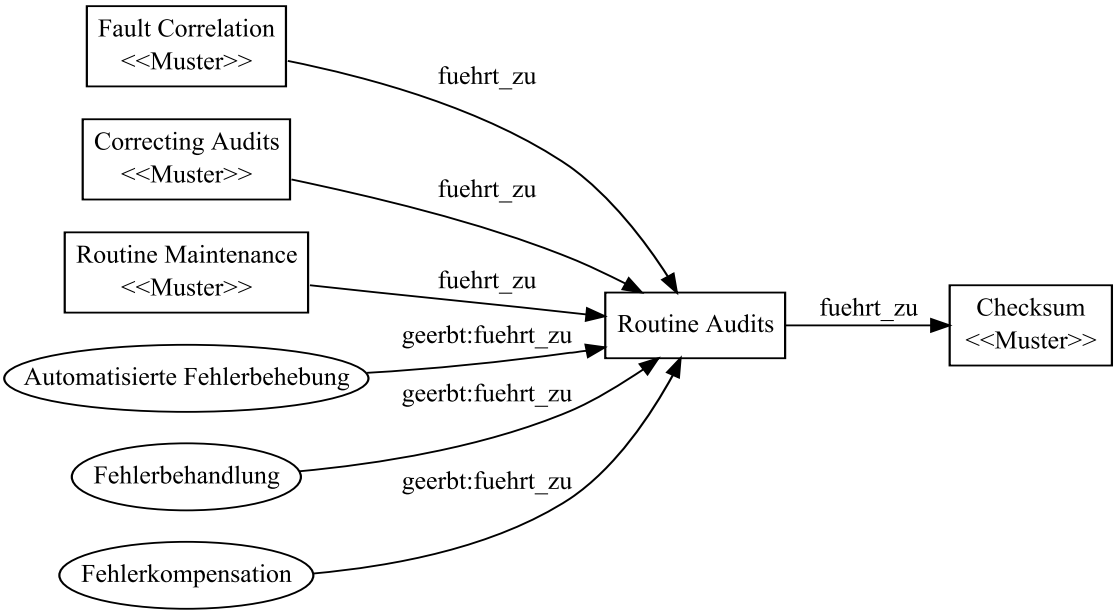
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Routine Audits" beschreibt einen Ansatz zur periodischen Validierung oder Kontrolle der Datenqualität.

Beziehungen - Grafik




Beziehungen - Detail

- **Fault Correlation -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Correcting Audits -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Routine Maintenance -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Routine Audits -> Checksum** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *176. Routine Exercises*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 124-126

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

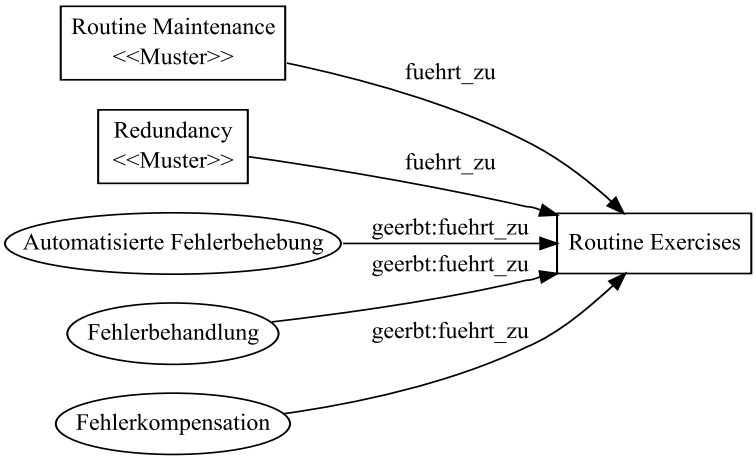
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Routine Exercises" beschreibt die Notwendigkeit die Funktionsfähigkeit von Fehler-Kompensationsmechnismen wie z. B. Failover-Mechanismen regelmäßig zu prüfen, sodass diese im Fehlerfall auch tatsächlich funktionieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Routine Maintenance -> Routine Exercises** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Redundancy -> Routine Exercises** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu

auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *177. Routine Maintenance*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 122-123

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

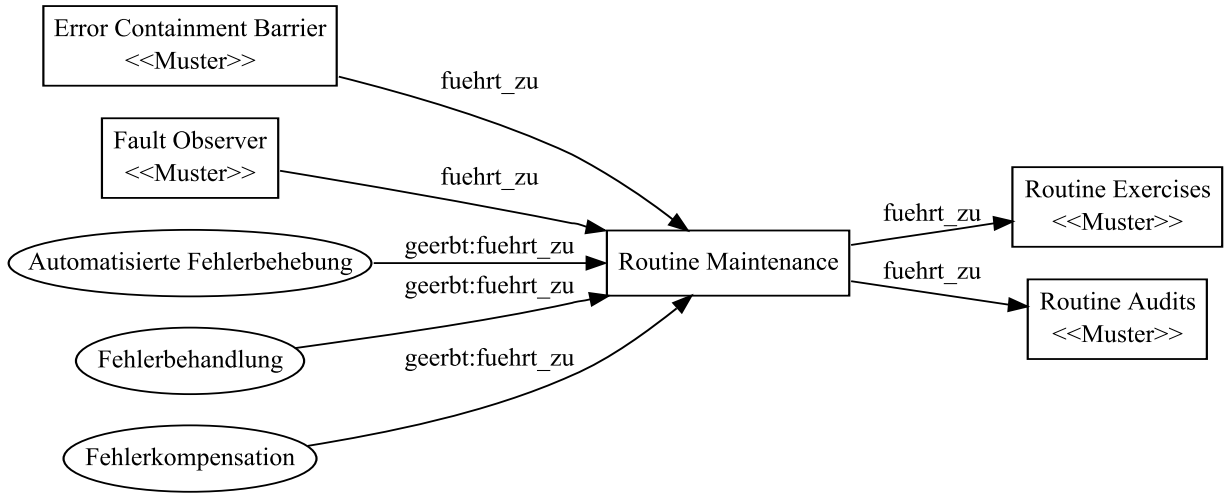
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Routine Maintenance" beschreibt den Zusammenhang zwischen regelmäßiger Wartung und der Funktionsfähigkeit bzw. Zuverlässigkeit eines Systems.

Beziehungen - Grafik




Beziehungen - Detail

- **Error Containment Barrier -> Routine Maintenance** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Fault Observer -> Routine Maintenance** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Routine Maintenance -> Routine Exercises** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Routine Maintenance -> Routine Audits** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *178. Rule based Transformation*

Schwerpunkt(e)

- Flexibilität

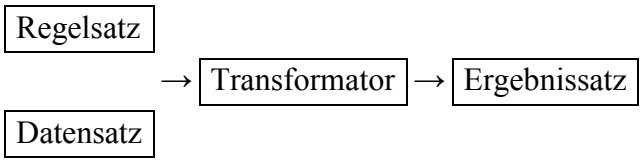
Gruppe(n)

- Datentransformation

Kontext
Strukturierte Daten werden an verschiedenen Stellen des Systems in unterschiedlicher Form benötigt. Ein Systembestandteil soll z. B. Dokumente verschiedener Standards oder verschiedener Formate, die den gleichen Inhaltstyp besitzen lesen und auf eine gemeinsame Art weiterverarbeiten können. Dazu wird eine flexible Form der Transformation benötigt, die auch eine einfache Anpassung der Transformation selbst, zur Erreichung unterschiedlicher Ergebnis-Datenstrukturen ermöglicht.

Problem
Zur Kommunikation mit unterschiedlichen Partnersystemen muss unter Umständen eine inhaltlich gleiche oder ähnliche Nachricht in unterschiedlichen Formaten übertragbar gemacht werden. Auch ist es möglich, dass eine Nachricht in mehrere Nachrichten gespalten oder mehrere Nachrichten zu einer Nachricht vereint werden müssen.

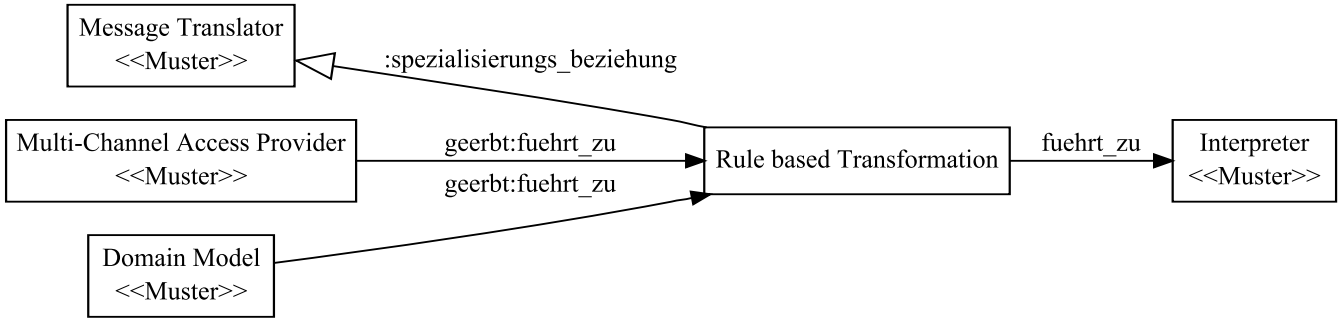
Lösung
Dieses Problem kann mittels regelbasierter Transformation von Nachrichten gelöst werden. Die Regeln (Regelsatz) sind hierbei eine Beschreibung der Transformation in einer domänen- oder problemspezifischen Sprache. Sie werden durch einen Interpreter (Transformator) eingelesen und auf den gegebenen Datensatz (ein oder mehrere zu transformierende Nachrichten) angewendet.




Beispiele
XSLT für die Transformation von XML-Dokumenten

SOA Design Patterns; Erl; S. 671 ff. und 681 ff. beschreibt mit den Pattern Data Model Transformation und Data Format Transformation zwei verwandte Pattern.

Beziehungen - Grafik




Beziehungen - Detail

- **Message Translator -> Rule based Transformation** :
  Die Rule based Transformation ist eine spezielle Ausprägung des Message-Translator-Patterns.
- **Rule based Transformation -> Interpreter** : Interpretation der Regeln
  Zur Interpretation von Transformationsregeln wird ein Interpreter benötigt.
- **Multi-Channel Access Provider -> Datentransformation** : Umsetzung von Nachrichten in unterschiedliche Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zur Umsetzung des Multi-Channel-Access-Provider-Pattern wird die Verwendung von mindestens einem Pattern der Gruppe Datentransformation benötigt, um die verschiedenen Zugriffsarten durch jeweils geeignet umgesetzte Nachrichten zu unterstützen.
- **Domain Model -> Datentransformation** : Wenn: Rich Domain Model
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: S. 117, Patterns of Enterprise Application Architecture, Folwer
  *"A rich Domain Model is better for more complex logic, but is harder to map to the database. A simple Domain Model can* [...]*, whereas a rich Domain Model requires Data Mapper."*

## *179. Scalability Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 241-246

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation
- Flexibilität

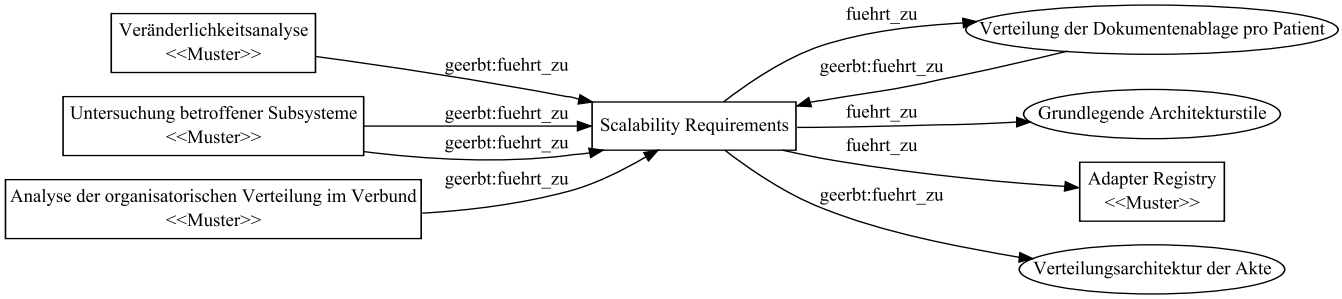
## Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen
- Kommunikationsanforderungen

## Kurzbeschreibung

Das Pattern "Scalability Requirements" beschreibt eine Darstellungsform für und einen Prozess zur Definition von Skalierbarkeits-Anforderungen. Skalierbarkeit beschreibt die Fähigkeit des Systems, in eine bestimmte Richtung zu wachsen, ohne dass relevante Teile des Systems neu entwickelt werden müssen.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Scalability Requirements -> Verteilung der Dokumentenablage pro Patient** : Skalierbarkeitsanforderungen beeinflussen Verteilung
  Die Patterns der Gruppe Verteilung der Dokumentenablage pro Patient beschreiben Architekturvarianten für die Gestaltung der Ablage von Dokumenten. Die zentrale Aufgabe einer verteilten Krankenakte beinhaltet den regelmäßigen schreibenden und lesenden Zugriff auf Dokumente zu Patienten. Die Skalierbarkeitsanforderungen beschreiben, in welcher Form und welchem Umfang das System wachsen und schrumpfen können soll. Diese Fähigkeit basiert unter anderem auf der Architektur der Dokumentenablage. Deshalb wird diese durch die Skalierbarkeitsanforderungen direkt beeinflusst.
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Scalability Requirements -> Adapter Registry** : macht Anzahl der Backends dyn. skalierbar
  Durch die Verwendung des Adapter-Registry-Musters kann die Anzahl von (auch verschiedenartigen) Backendsystemen dynamisch erweitert oder verkleinert werden, ohne den Code des Systems zu verändern. Je nach Implementierung ist auch eine Veränderung ohne Neustart möglich.

- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *180. Schichtenarchitektur*

Englische Bezeichnung

- Layers

Quellen
**POSA 1, A System of Patterns, Buschmann**
S. 31-51

Pattern-Oriented Software Architecture 1, A System of Patterns, Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Wiley Series in Software Design Patterns, 1996

### Schwerpunkt(e)

- Flexibilität

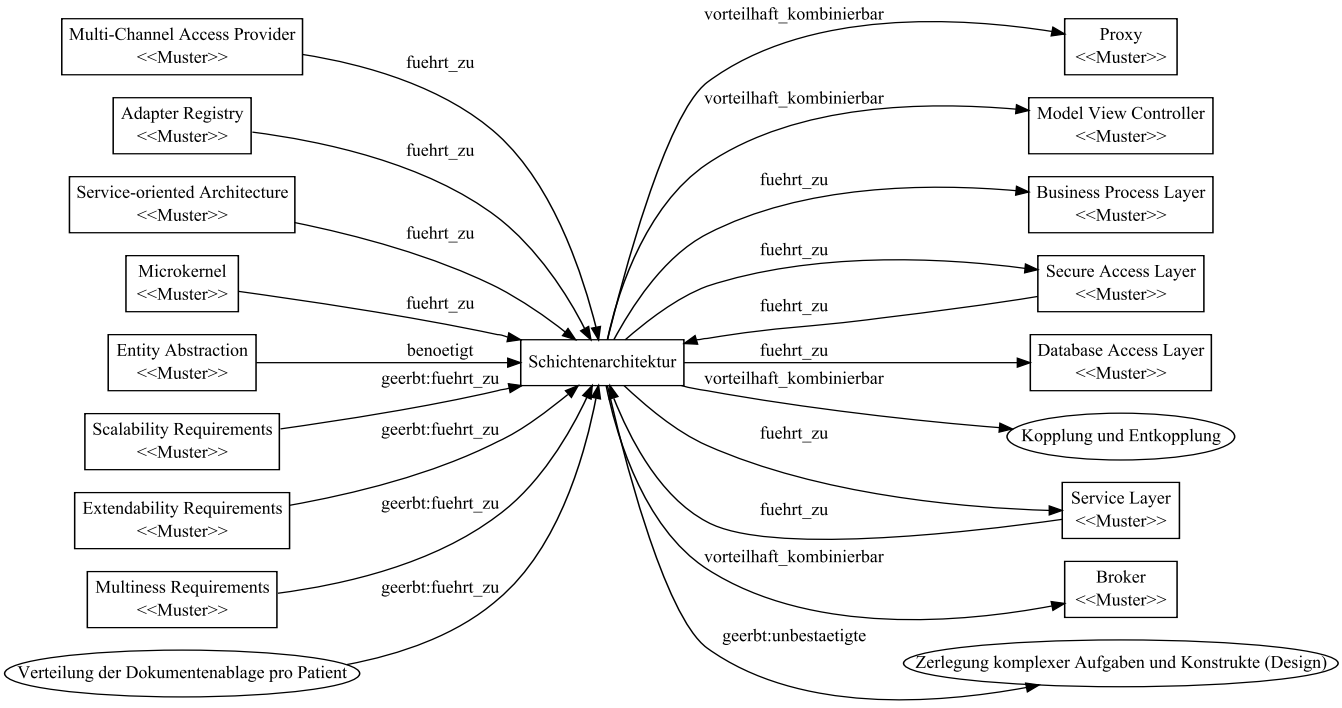
### Gruppe(n)

- Grundlegende Architekturstile
- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

### Kurzbeschreibung

Das Pattern Schichtenarchitektur beschreibt die Zerlegung einer Softwarearchitektur in logische Schichten, wobei die Bestandteile einer Schicht immer nur mit Bestandteilen der sie direkt umgebenden beiden Schichten interagieren. Ein einzelne Schichten überspringender Zugriff widerspricht der Schichtenarchitektur.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Service Layer -> Schichtenarchitektur** :
  Die Service Layer ist eine spezielle Schicht. Sie benötigt die Schichtenarchitektur als Umfeld, um angewendet werden zu können.
- **Multi-Channel Access Provider -> Schichtenarchitektur** : Als Bestandteil
  Das Multi-Channel-Access-Provider-Pattern ist selbst in Form einer

Schichtenarchitektur aufgebaut. Es besteht aus den Schichten Basisschicht und Schnittstellenschicht.

- **Adapter Registry -> Schichtenarchitektur** : Interne Gliederung in Schichten
  Das Adapter-Registry-Pattern ist intern in die Schichten Registry-Schicht und Adapter-Schicht gegliedert.
- **Service-oriented Architecture -> Schichtenarchitektur** : Basiert auf einer Schichtenarchitektur
  Service orientierte Architekturen sind immer in Schichten organisiert.
  Quelle: SOA in Practice, Josuttis; S. 109-117
  Alle dargestellten SOA-basierten Architekturmodelle basieren auf dem Schichtenarchitektur-Muster.
- **Microkernel -> Schichtenarchitektur** : zur Gliederung in Schichten
  Es ist häufig sinnvoll, das Microkernel-Pattern intern in Schichten von Erweiterungen zu gliedern. Dennoch entsteht daraus eine Sonderform der Schichtenarchitektur, die sich durch ihre erhöhte Flexibilität von vielen anderen Formen der Umsetzung von Schichtenarchitekturen unterscheidet.
  Quelle: S. 192 in Pattern oriented Software Architecture, Band 1.
  Übersetzt ins Deutsche und sinngemäß wiedergegeben: *„Die Beziehung zwischen dem Schichtenarchitektur- und Microkernel-Pattern ist zwiespältig. Einerseits kann das Mikrokernel-Pattern als Variante des Schichtenarchitektur-Patterns verstanden werden* [...] *Andererseits können beide Patterns in manchen Anwendungsdomänen auch alternativ verwendet werden"*
- **Secure Access Layer -> Schichtenarchitektur** :
  Wenn eine Secure Access Layer verwendet werden soll, muss die so gestaltete Software aus mindestens einer weiteren Schicht bestehen und ist folglich in einer Schichtenarchitektur organisiert. Dabei muss die Schichtenarchitektur nicht zwangsläufig das einzige verwendete oder das dominierende Architektur-Pattern sein.
- **Schichtenarchitektur -> Proxy** :
  Generell kann die Kopplung zwischen den Schichten einer Schichtenarchitektur durch die Kappselung der Schnittstelle der jeweils konsumierten Schicht mit Stellvertreterobjekten (Proxys) verringert werden. Werden die Schichten der Architektur in einem verteilten Szenario betrieben, so ist die Verwendung von Remote Proxys zwischen den Schichten sogar notwendig um die Kommunikation zu verbergen.
- **Schichtenarchitektur -> Model View Controller** : GUI-Schicht
  Ist ein System in Form einer Schichtenarchitektur aufgebaut, so kann die Schicht seiner Benutzeroberfläche intern nach dem Model-View-Controller-Pattern strukturiert sein. Die Existenz einer Schichtenarchitektur ist allerdings keine notwendige Voraussetzung für die Verwendung von Model View Controller.
- **Schichtenarchitektur -> Business Process Layer** : wenn prozessorientierter Anwendungsbereich
  Bei Anwendungssystemen, deren Aufgabe die Abbildung von einem oder mehreren Prozessen ist, kann durch die Definition einer eigenen Business Process Layer (Geschäftsprozess-Schicht) die Aufgabe der Abbildung des Prozessflusses in einer

von anderen Aufgaben der Anwendung entkoppelten Form realisiert werden. Sie eignet sich deshalb besonders für häufiger veränderliche Prozesse.

- **Schichtenarchitektur -> Secure Access Layer** : wenn sicherheitskritische Remotezugriffe
  Wenn auf eine darunterliegende Anwendungsschicht (nach dem Schichtenarchitektur-Muster strukturiert) zugegriffen werden soll und die Operationen möglicherweise sicherheitskritisch sind, empfiehlt sich die Einführung einer gesicherten Zugriffsschicht.
- **Schichtenarchitektur -> Database Access Layer** : Zugriffschicht für Datenquelle(n)
  In einer Schichtenarchitektur kann eine spezielle Schicht für den Zugriff auf ein oder mehrere Datenquellen eingeplant werden. Das ist vor allem dann nützlich, wenn der Zugriff auf diese Datenquelle(n) die Komplexität des Quellcodes so erhöhen würde, dass dieser dadurch weniger übersichtlich wird.
- **Schichtenarchitektur -> Kopplung und Entkopplung** : Entkopplung der Schichten
  Eine Schichtenarchitektur ist vorteilhaft mit Patterns der Gruppe Kopplung und Entkopplung, wie z. B. dem Proxy-Pattern und dem Adapter-Pattern kombinierbar. Diese Patterns können verwendet werden um die einzelnen Schichten einer Schichtenarchitektur stärker zu entkoppeln.
- **Schichtenarchitektur -> Service Layer** : Eine mögliche Schicht
  Das Service-Layer-Pattern beschreibt eine (meist) remotefähige Schicht, die innerhalb einer Schichtenarchitektur angewendet werden kann.
- **Schichtenarchitektur -> Broker** :
  Siehe Pattern Oriented Software Architecture, Volume 4, S. 238 : *"Most Broker Realizations are based on a Layers Architecture* [...]*"*
- **Scalability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Skalierbarkeit. Die Fähigkeit zum Betrieb im Cluster, die Nutzung mehrerer CPUs usw. sind direkt von der Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Skalierbarkeitsanforderungen (Scalability Requirements) die Auswahl von Architektur-Patterns.
- **Extendability Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Softwarearchitektur einer Anwendung beeinflusst ihre Erweiterbarkeit. Der Aufwand, neue Funktionen zu einem bestehenden System hinzuzufügen ist direkt von dessen Softwarearchitektur abhängig. Aus diesem Grund beeinflussen die Extendability Requirements die Auswahl von Architektur-Patterns.
- **Multiness Requirements -> Grundlegende Architekturstile** : Beeinflusst die Auswahl von Architekturmustern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Multiness-Requirements-Pattern wird angewendet, um zu spezifizieren, wie ein System die Dienste mehrerer unterschiedlicher Systeme/Standards usw. zur gleichen

Zeit anbieten muss. Ein System, das von anderen Systemen oder Nutzern als entsprechend vielgesichtig wahrgenommen wird, kann das nur durch eine entsprechende Softwarearchitektur (Beispiel: Multi-Channel-Access-Provider-Pattern) erreichen.

- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *181. Secure Access Layer*

Quellen

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

S. 24 ff.

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

Schwerpunkt(e)

- Sicherheit

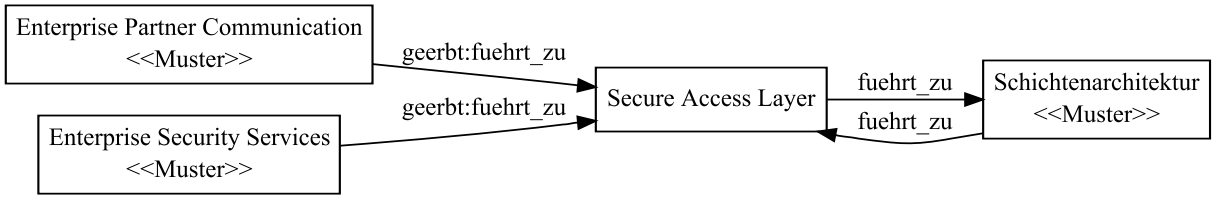
Gruppe(n)

- Point of Access

Kurzbeschreibung

Das Pattern "Secure Access Layer" beschreibt einen Ansatz der auf einer zusätzlichen Schicht zur Erreichung sicheren Zugriffs auf Inhalte oder Funktionen einer Anwendung basiert.

Beziehungen - Grafik




Beziehungen - Detail

- **Schichtenarchitektur -> Secure Access Layer** : wenn sicherheitskritische Remotezugriffe
  Wenn auf eine darunterliegende Anwendungsschicht (nach dem Schichtenarchitektur-Muster strukturiert) zugegriffen werden soll und die Operationen möglicherweise sicherheitskritisch sind, empfiehlt sich die Einführung einer gesicherten Zugriffsschicht.
- **Secure Access Layer -> Schichtenarchitektur** :
  Wenn eine Secure Access Layer verwendet werden soll, muss die so gestaltete Software aus mindestens einer weiteren Schicht bestehen und ist folglich in einer Schichtenarchitektur organisiert. Dabei muss die Schichtenarchitektur nicht zwangsläufig das einzige verwendete oder das dominierende Architektur-Pattern sein.
- **Enterprise Partner Communication -> Point of Access** : Gestaltung des Zugriffspunkts für Partner
  Geerbte Beziehung: fuehrt_zu_beziehung.
  Das Enterprise-Partner-Communication-Pattern beschreibt die Kommunkations-anforderungen zwischen Organisationen mit voneinander getrennt existierender IT-Infrastruktur. Zur Umsetzung der Enterprise Partner Communication muss ein Muster für die Gestaltung des Point of Access ausgewählt werden, um für den Kommunikationspartner einen Zugangsweg zu dem betreffenden IT-System zu eröffnen.
- **Enterprise Security Services -> Point of Access** : Gestaltung eines sicheren Zugriffspunkts für interne Applikationen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um einen sicheren Zugriffsweg auf interne Applikationen zu gestalten sollte ein Pattern der Gruppe Point of Access angewendet werden.

## *182. Security Accounting Requirements*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 360-368

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

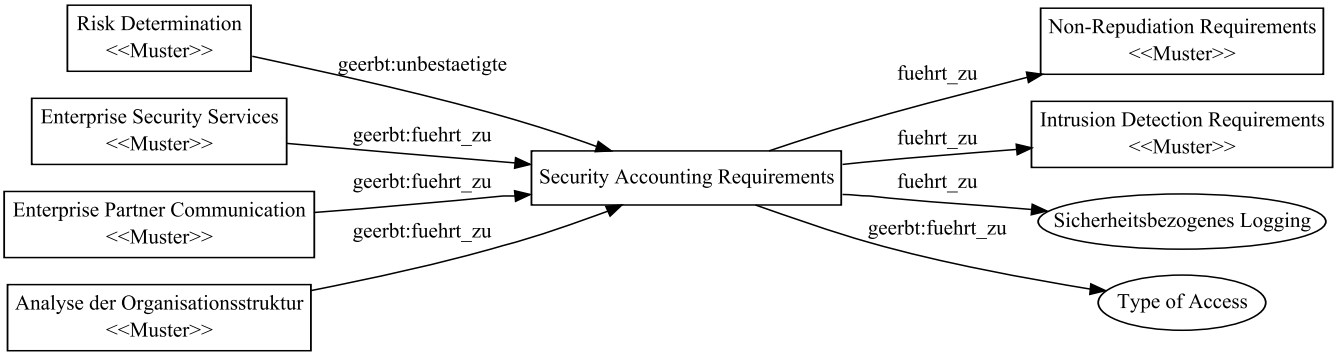
Gruppe(n)

- Ermittlung der Sicherheitsanforderungen

Kurzbeschreibung

Das Pattern "Security Accounting Requirements" beschreibt ein Verfahren zur Spezifikation von Anforderungen an eine dem jeweiligen Problem angemessene sicherheitsbezogene Protokollierung von Ereignissen.

Beziehungen - Grafik




Beziehungen - Detail

- **Security Accounting Requirements -> Non-Repudiation Requirements** : Wenn verschärfte Anforderungen an Beweissicherheit vorliegen
  Quelle: Security Patterns, Schumacher et al.; S. 396 u. 360 (kombiniert betrachtet)
- **Security Accounting Requirements -> Intrusion Detection Requirements** : Einbruchserkennung
  Die Anforderungsdefinition für eine Einbruchserkennung setzt die Kenntnis der Daten über erfasste sicherheitsrelevante Vorfälle voraus.
  Quelle: Security Patterns; Schumacher et al.; S. 388; Context
- **Security Accounting Requirements -> Sicherheitsbezogenes Logging** : umsetzbar durch
  Die einzelnen Security Accounting Requirements können durch verschiedene Logging-Mechanismen umgesetzt werden.
- **Risk Determination -> Ermittlung der Sicherheitsanforderungen** : Input: Bewertete Risiken
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Menge der durch die Anwendung des Risk-Determination-Patterns ermittelten gewichteten und bewerteten Risiken sollte auch bei der detaillierten Spezifikation von Sicherheitsanforderungen berücksichtigt werden. Nachgewiesen sind die Beziehungen

zu den Mustern, die in der Abfolge der Ermittlung der Sicherheitsanforderungen übergeordnet sind.

- **Enterprise Security Services -> Ermittlung der Sicherheitsanforderungen** :
  Informationsbasis
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: Security Patterns, Schumacher et al.; S. 61; Absatz zu Enterprise Security Services: „*Primäre Beispiele für solche Dienste* [Enterprise Security Services] *sind Identifizierung und Authentifizierung, Accounting und Auditing, Zugriffskontrolle und Autorisierung und Sicherheitsmanagement.“*
  Die zitierte Aussage zeigt, dass ermittelte Enterprise Security Services in den verschiedenen genannten Gebieten, die einen großen Teil der Patterns der Gruppe Ermittlung der Sicherheitsanforderungen ausmachen, durch Muster der referenzierten Gruppe feinspezifiziert und dokumentiert werden.
- **Enterprise Partner Communication -> Ermittlung der Sicherheitsanforderungen** : Input: Arten der Kommunikation mit Partnern
  Geerbte Beziehung: fuehrt_zu_beziehung
  Notwendige Kommunikation mit Geschäftspartnern oder vertraglich verbundenen Gesundheitsdienstleistern beeinflusst die Sicherheitsanforderungen, die an ein IT-System zur verteilten Abbildung von Krankenakten zu stellen sind. So sind durch die zusätzlichen Akteure "Geschäftspartner" ggf. zusätzliche Rollen, zusätzliche Zugriffspunkte usw. notwendig. Daraus resultiert die Notwendigkeit zusätzliche Requirements für diese zusätzlichen Elemente zu spezifizieren.
- **Analyse der Organisationsstruktur -> Ermittlung der Sicherheitsanforderungen** :
  Input: Aufgaben, Rollen und Akteure
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anwendung des Analyse-der-Organisationsstruktur-Patterns liefert als Ergebnis Rollen und Akteure, die den Aufgaben aus dem Analyse-der-betroffenen-Behandlungspfade-Pattern zugeordnet sind. Diese Informationen können als Basis für die Definition verschiedener Sicherheitsanforderungen (vor allem aus dem Bereich Autorisierung) verwendet werden.
- **Ermittlung der Sicherheitsanforderungen -> Type of Access** :
  Sicherheitsanforderung als Auswahlkriterien
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Sicherheitsanforderungen, insbesondere I&A Requirements, Access Control Requirements und Roles, dienen zur Auswahl des geeigneten Type of Access. Beispiel: Es eignet sich beispielsweise der Typ Full Access with Errors immer dann nicht, wenn eine große Menge von Rollen mit sehr unterschiedlichen Berechtigungen dazu führt, dass bei einem Großteil der angezeigten Funktionen dem Benutzer die Ausführung der Funktion mit einem Berechtigungsfehler untersagt wird. In diesem Fall ist die Wahl des Typs Limited Access besser geeignet, da hier nur Funktionen für die der Benutzer eine Berechtigung besitzt, angeboten werden.

## *183. Security Needs Identification for Enterprise Assets*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 89-102

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

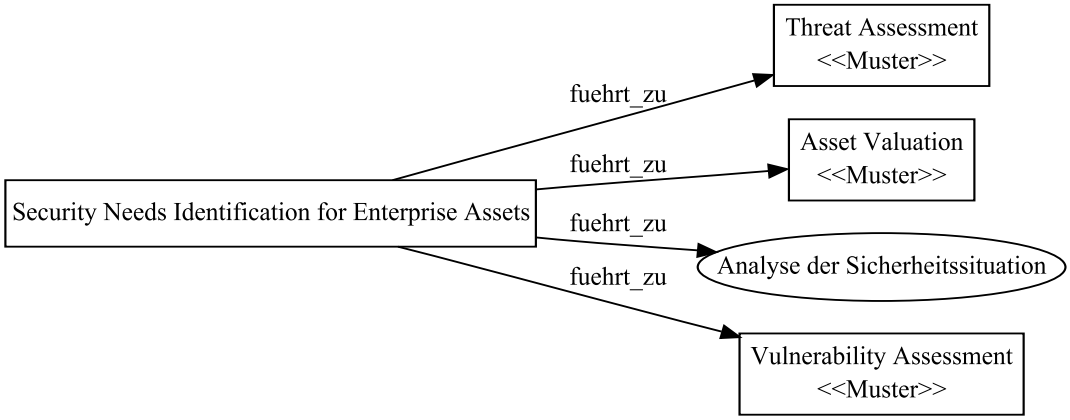
Gruppe(n)

- Besitzt als Einstiegspattern in den Schwerpunkt Sicherheit keine Gruppenzuordnung

Kurzbeschreibung

Das Pattern "Security Needs Identification for Enterprise Assets" ist das Einstiegs-Pattern in die Analyse der Sicherheitsanforderungen. Sein Ergebnis ist ein erster Überblick über die relevanten, zu sichernden Aspekte.

Beziehungen - Grafik




Beziehungen - Detail

- **Sicherheit -> Security Needs Identification for Enterprise Assets** : Einstieg in die Anforderungsanalyse bezüglich Sicherheit
  Das Pattern Security Needs Identification for Enterprise Assets ist direkt dem Schwerpunkt Sicherheit zugeordnet. Es bildet das Einstiegspattern in die Analyse des Sicherheitsbedarfs und die anschließende Konzeption einer geeigneten Sicherheitslösung.
- **Security Needs Identification for Enterprise Assets -> Threat Assessment** : Analyse der Bedrohungssituation
  Quelle: Security Patterns; Schumacher et al.; S. 60

- **Security Needs Identification for Enterprise Assets -> Asset Valuation** :
  Bewertung bestehender Infrastruktur
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Security Needs Identification for Enterprise Assets -> Analyse der Sicherheitssituation** : Grobanalyse zu Detailanalyse
  Das Security-Needs-Identification-for-Enterprise-Assets-Pattern dient der Identifikation und Bewertung gefährdeter Güter, Anlagen, Personen und Daten in den untersuchten medizinischen Versorgungseinrichtungen. Die ermittelten Gefährdeten, sowie das ebenfalls ermittelte Einflusspotenzial auf den Gesamtbetrieb dienen dazu im Rahmen der Analyse der Sicherheitssituation ermittelte Sicherheitsrisiken und andere Einflussfaktoren nach ihrer Wichtigkeit zu bewerten.
- **Security Needs Identification for Enterprise Assets -> Vulnerability Assessment** :
  Identifikation von Schwächen
  Quelle: Security Patterns; Schumacher et al.; S. 60

## *184. Security Session*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 297-304

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

S. 14 ff. als Session

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

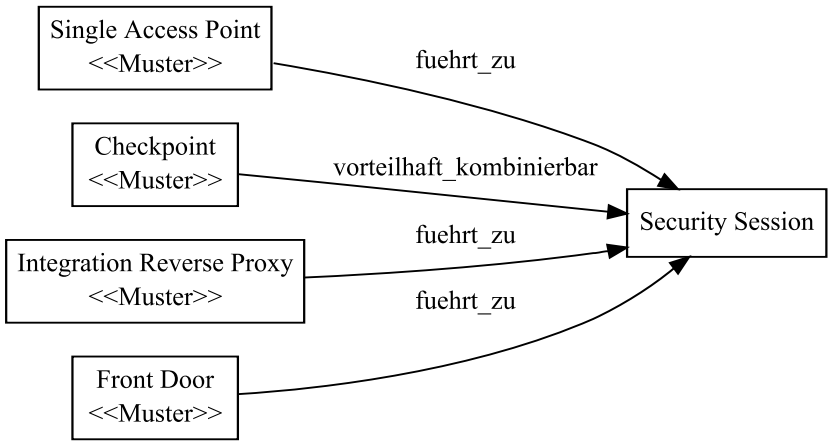
Schwerpunkt(e)

Gruppe(n)

Kurzbeschreibung

Das Pattern "Security Session" beschreibt den Ansatz, ein Sitzungs-Objekt für jeden aktuell angemeldeten Benutzer, also jede Benutzersitzung zu führen. Über dieses Sitzungs-Objekt sind die sicherheitsrelevanten Informationen zu dem jeweiligen Benutzer und zur Sitzung verfügbar.

Beziehungen - Grafik




Beziehungen - Detail

- **Single Access Point -> Security Session** : day pass
  Quelle: Security Patterns; Schumacher et al; S. 284
  Zweck: Erhalt der beim Checkpoint ausgestellten Identitäts- und Berechtigungsinformation
- **Checkpoint -> Security Session** : typically relies on
  Quelle: Security Patterns; Schumacher et al.; S. 304

Quelle: S. 299; Security Patterns: *"A systems Checkpoint is the usual Place to instantiate the session object and set up its initial values."*

- **Integration Reverse Proxy -> Security Session** : relies on
  Quelle: Security Patterns; Schumacher et al.; S. 304
- **Front Door -> Security Session** : Verwendet
  Quelle: Security Patterns; Schumacher et al.; S. 304;
  *"Integration Reverse Proxy and Front Door rely on Security Session to keep track of Web users."*

## *185. Service Layer*

Quellen

**Patterns of Enterprise Application Architecture, Fowler**

S. 133-142

Martin Fowler, Patterns of Enterprise Application Architecture, 1. Aufl. (Addison-Wesley Professional, 2002).

Schwerpunkt(e)

- Flexibilität

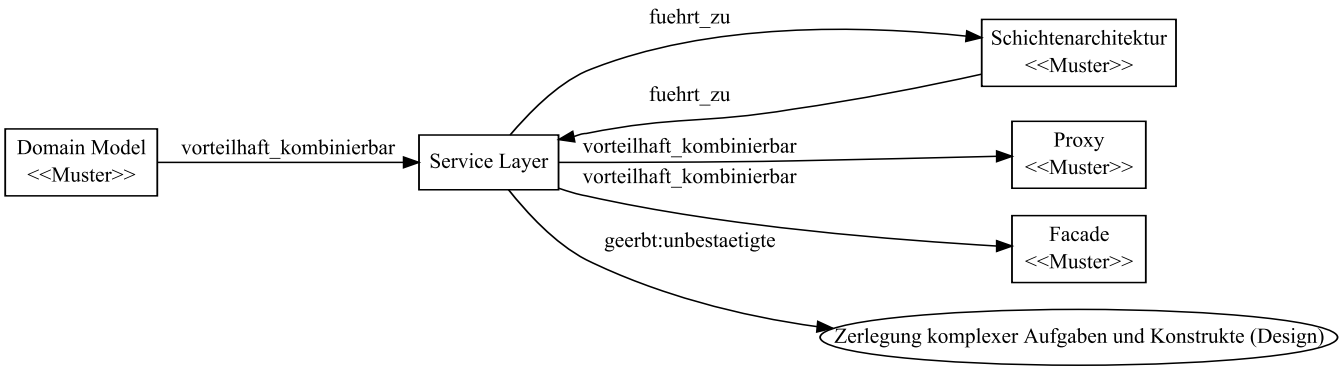
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Architektur)

Kurzbeschreibung

Das Pattern "Service Layer" beschreibt eine Schicht, die Anwendungslogik als Dienste (z. B. für den Zugriff auf Bestandteile des Domain Model) kapselt.

Beziehungen - Grafik




Beziehungen - Detail

- **Schichtenarchitektur -> Service Layer** : Eine mögliche Schicht
  Das Service-Layer-Pattern beschreibt eine (meist) remotefähige Schicht, die innerhalb einer Schichtenarchitektur angewendet werden kann.

- **Domain Model -> Service Layer** : Konkrete Dienste zur Nutzung des Domain-Model
  Quelle: S. 119, Patterns of Enterprise Application Architecture, Folwer;
  *"Whe you use Domain Model you may want to consider Service Layer to give your Domain Model a more distinct API."*
- **Service Layer -> Schichtenarchitektur** :
  Die Service Layer ist eine spezielle Schicht. Sie benötigt die Schichtenarchitektur als Umfeld, um angewendet werden zu können.
- **Service Layer -> Proxy** :
  Um die Kopplung zwischen der konkreten Implementierung der Dienste der Service-Layer und den konsumierenden Softwarebestandteilen der darüberliegenden Schicht zu verringern, kann die Schnittstelle der Service Layer durch Stellvertreter-Objekte (Proxys) gekapselt werden.
- **Service Layer -> Facade** : Als Remote Facade
  Quelle: Patterns fo Enterprise Application Architecture; Fowler; S. 135
  *"Add remotability when you need it (if ever) by putting Remote Facades (388) on your Service Layer* [...]*"*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *186. Service-oriented Architecture*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 8

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

**SOA in Practice, Josuttis**

S. 1 - 284; Umfassende Beschreibung von SOA, allerdings nicht in Patternform. Dient vor allem als Quelle für Beziehungen.

SOA in Practice, The Art of Distributed System Design; Nicolai M. Josuttis; O´Reilly 2007

Schwerpunkt(e)

- Kommunikation

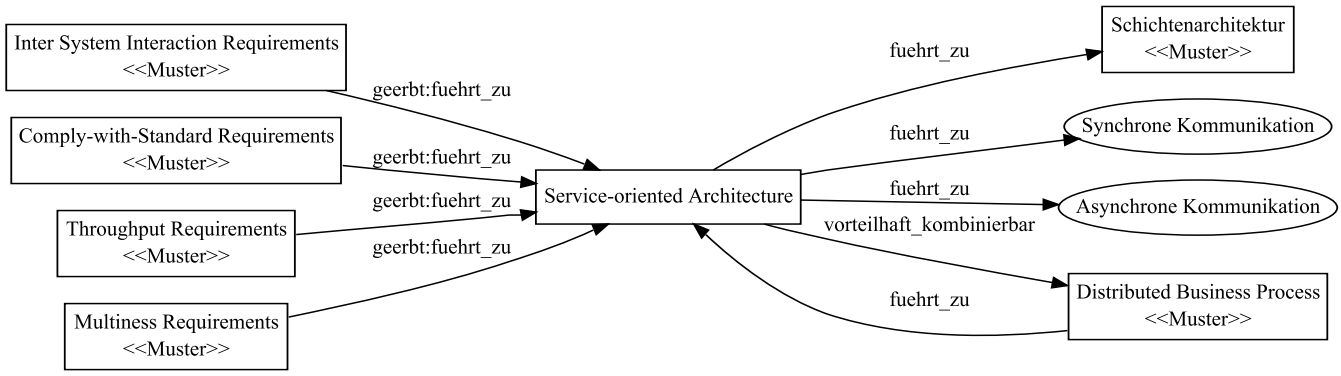
### Gruppe(n)

- Grundlegende Integrationsformen

### Kurzbeschreibung

Das Pattern "Service-oriented Architecture" beschreibt eine Architekturform, die auf der Verwendung entfernt aufrufbarer Dienste basiert. Sie hat die Integration aller verfügbaren Dienste einer Einrichtung zu einem Pool von Diensten, auf deren Basis wiederum die Frontends verschiedener Anwendungsprogramme aufgebaut werden können zum Ziel.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Distributed Business Process -> Service-oriented Architecture** : Benötigt SOA oder Shared Business Functions
  Um einen oder mehrere verteilte Geschäftsprozesse umzusetzen wird als Basis die Existenz von gemeinsam nutzbaren Anwendungsfunktionen (Shared Business Functions) oder noch umfassender, die Existenz einer Service orientierten Architektur(SOA) benötigt.
- **Service-oriented Architecture -> Schichtenarchitektur** : Basiert auf einer Schichtenarchitektur
  Service orientierte Architekturen sind immer in Schichten organisiert.
  Quelle: SOA in Practice, Josuttis; S. 109-117
  Alle dargestellten SOA-basierten Architekturmodelle basieren auf dem Schichtenarchitektur-Muster.
- **Service-oriented Architecture -> Synchrone Kommunikation** : für synchrone Dienste
  Quelle: SOA in Practice, Josuttis; S. 124; Message Exchange Pattern: Request/Response;
  Das in der Quelle dargestellte Message-Exchange-Pattern beschreibt einen synchronen RPC-ähnlichen Aufruf durch Request/Reply Kommunikation.
- **Service-oriented Architecture -> Asynchrone Kommunikation** : für asynchrone Dienste
  Quelle: SOA in Practice, Josuttis; S. 125; Message Exchange Pattern: One-Way
  Quelle: SOA in Practice, Josuttis; S. 126; 10.2.3 Request/Response Versus Two One-

Way Messages
Quelle: SOA in Practice, Josuttis; S. 128; Message Exchange Pattern: Request/Callback
Quelle: SOA in Practice, Josuttis; S. 129; Message Exchange Pattern: Publish/Subscribe

- **Service-oriented Architecture -> Distributed Business Process** : Ergänzen einander sinnvoll
  Quelle: Enterprise Integration Patterns, Hohpe, S. 9 kurz vor Abbildung zu Business-to-Business Integration
  Quelle: SOA in Practice, Josuttis; S. 72; Process-Enabled SOA
  Fazit: Die Prinzipien serviceorientierter Architekturen sind geeignet die Abbildung von verteilten Geschäftsprozessen zu unterstützen.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.
- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## *187. Share the Load*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 206-207

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

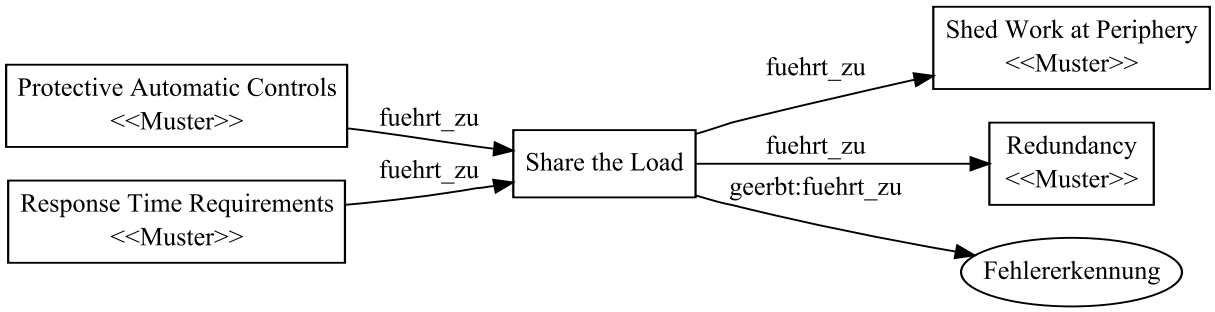
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Share the Load" beschreibt den verbreiteten Ansatz, hohe Last zwischen verschiedenen redundanten Knoten zu verteilen.

Beziehungen - Grafik




Beziehungen - Detail

- **Protective Automatic Controls -> Share the Load** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Response Time Requirements -> Share the Load** : umsetzbar durch
  Um auch bei Lasten, die die Leistungsfähigkeit eines einzelnen Knotens überschreiten, die Response Time Requirements einzuhalten, kann die Last zwischen redundanten Knoten verteilt werden. Diese Lösung setzt eine gleichzeitige Verwendung des Redundancy Patterns voraus.
- **Share the Load -> Shed Work at Periphery** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Share the Load -> Redundancy** : Erreichung von Gleichen, die sich die Last teilen können
  Um das Share-the-Load-Pattern anzugwenden ist es notwendig, dass redundante

Systembestandteile existieren, zwischen denen die Last verteilt werden kann.
Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 206

- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *188. Shared Business Function*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 7

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

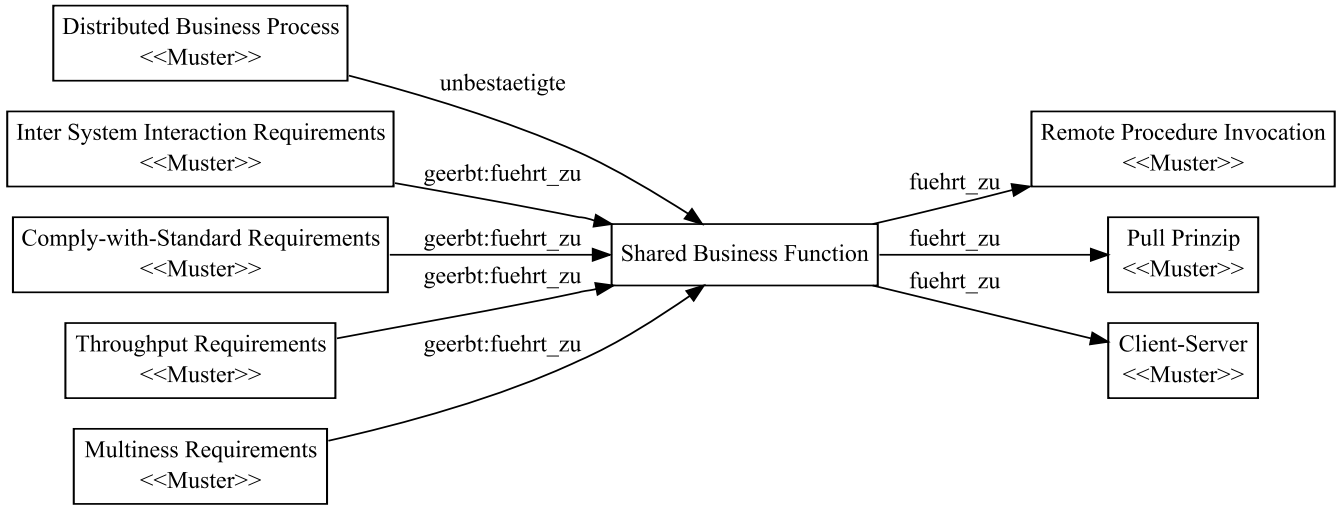
Gruppe(n)

- Grundlegende Integrationsformen

Kurzbeschreibung

Das Pattern "Shared Business Function" beschreibt eine Integrationsform, die auf der gemeinsamen Nutzung von (fachlich bedingten und zumeist remotefähigen) Funktionen basiert.

Beziehungen - Grafik



Beziehungen - Detail

- **Distributed Business Process -> Shared Business Function** : Benötigt SOA oder Shared Business Functions
  Um einen oder mehrere verteilte Geschäftsprozesse umzusetzen wird als Basis die Existenz von gemeinsam nutzbaren Anwendungsfunktionen (Shared Business Functions) oder noch umfassender, die Existenz einer Service orientierten Architektur(SOA) benötigt.
- **Shared Business Function -> Remote Procedure Invocation** : umsetzbar durch Erschließt sich aus dem Text von S. 7 in Enterprise Integration Patterns; Hohpe.
- **Shared Business Function -> Pull Prinzip** : Übliches Kommunikationsprinzip Anwendungen, die auf Shared Business Functions aufbauen, sind bezogen auf ihre hauptsächliche Kommunikationsrichtung häufig nach dem Pull-Prinzip aufgebaut. Das bedeutet, dass Daten nicht an alle später möglichen Konsumenten verteilt werden, sondern über eine gemeinsame Funktion (Shared Business Function) bei Bedarf abgerufen werden.
- **Shared Business Function -> Client-Server** : besitzt eindeutige Anbieter/Konsument Beziehung
  Das Shared-Business-Function-Pattern beschreibt sehr allgemein die gemeinsame Nutzung von fachlichen Funktionen in verschiedenen Anwendungen eines Unternehmens oder im vorliegenden Fall in medizinischen Versorgungseinrichtungen. Bietet ein System anderen Systemen eine Teilmenge seiner Funktionen zur gemeinsamen Nutzung an, so tritt dieses System eindeutig in der Rolle des Servers auf, während alle Systeme, die diese Funktionen nutzen die Rolle von Clients einnehmen.
- **Inter System Interaction Requirements -> Grundlegende Integrationsformen** : Auswahl geeigneter Integrationsform
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Anforderungen der Inter System Interaction Requirements bilden die Grundlage für die Auswahl einer geeigneten Integrationsform
- **Comply-with-Standard Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards definieren häufig, wie die Kommunikation zwischen den beteiligten Knoten stattfinden soll. Das gilt vor allem auch für medizinische Standards wie z. B. DICOM, HL7 etc. Deshalb beeinflussen Anforderungen, einen Standard zu Implementieren, die Auswahl der Integrationsform durch eine Einschränkung der Menge auswählbarer Formen.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.

- **Multiness Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Wahl der Integrationsform/Integrationsarchitektur beeinflusst, ob die Anforderungen an die "Vielgesichtigkeit" der Anwendung erfüllt werden können. Formen wie z. B. die Data-Replication führen zu einer größeren Zahl von, ausschließlich durch die Replikation von Daten gekoppelten Subsystemen. Diese Subsysteme mit einer Vielzahl von Schnittstellen und Schnittstellen-Standards auszustatten ist deutlich aufwendiger, als das z. B. bei einer SOA oder einem Information Portal der Fall ist.

## *189. Shared Database*

Quellen

**Enterprise Integration Patterns, Hohpe**

S. 47

Enterprise Integration Patterns, Designing, Building, and Deploying Messaging Solutions, Gregor Hohpe and Bobby Woolf, Addison Wesley Signature Series, 14. Auflage Mai 2010,

Schwerpunkt(e)

- Kommunikation

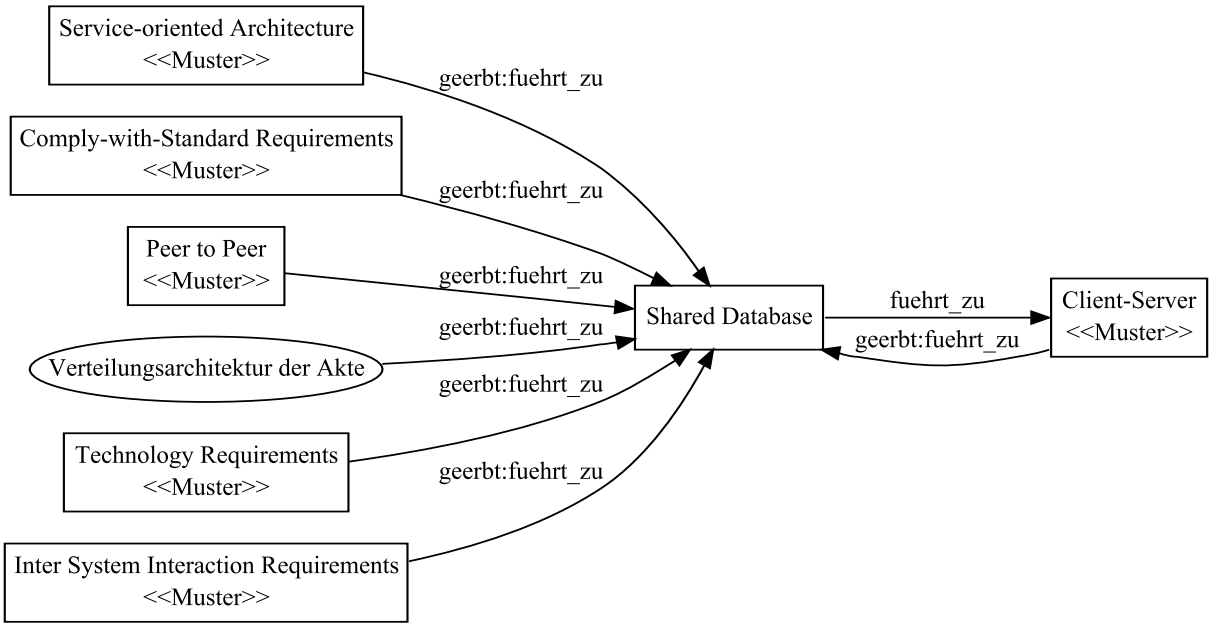
Gruppe(n)

- Synchrone Kommunikation

Kurzbeschreibung

Das Pattern "Shared Database" beschreibt den Ansatz, eine gemeinsame Datenbank zu verwenden, um die Daten verschiedener Anwendungen zu integrieren.

Beziehungen - Grafik




Beziehungen - Detail

- **Shared Database -> Client-Server** :
  Systeme, die **direkt** auf der Verwendung einer gemeinsamen Datenbank aufbauen sind Client-Server-Architekturen, da die Datenbank die Rolle eines gemeinsamen Servers einnimmt und in keiner zu erwartenden Kommunikationssituation auch als Client aktiv wird.
- **Service-oriented Architecture -> Synchrone Kommunikation** : für synchrone Dienste
  Geerbte Beziehung: fuehrt_zu_beziehung
  Quelle: SOA in Practice, Josuttis; S. 124; Message Exchange Pattern: Request/Response;
  Das in der Quelle dargestellte Message-Exchange-Pattern beschreibt einen synchronen RPC-ähnlichen Aufruf durch Request/Reply Kommunikation.
- **Comply-with-Standard Requirements -> Synchrone Kommunikation** : wenn im Standard gefordert
  Geerbte Beziehung: fuehrt_zu_beziehung
  Standards, deren Verwendung in den Anforderungen gefordert werden, können die Auswahl des Kommunikationsmechanismus festlegen.
- **Peer to Peer -> Synchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Peer-to-Peer-Kommunikation ist mit mitteln synchroner Kommunikation realisierbar.
- **Client-Server -> Synchrone Kommunikation** : realisierbar mit
  Geerbte Beziehung: fuehrt_zu_beziehung
  Client-Server-Systeme sind mit Mitteln synchroner Kommunikation realisierbar
- **Verteilungsarchitektur der Akte -> Synchrone Kommunikation** : Auswahl von Kommunikationsmechanismen
  Geerbte Beziehung: fuehrt_zu_beziehung

Abhängig von der Beschreibung der Ausprägung des ausgewählten Verteilungsarchitektur-Patterns werden ein oder mehrere Mechanismen synchroner und asynchroner Kommunikation benötigt um die Verteilungsarchitektur umzusetzen.

- **Technology Requirements -> Synchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Inter System Interaction Requirements -> Synchrone Kommunikation** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Beschreibung der Interaktion zwischen Systemen an einer Schnittstelle ermöglicht die Auswahl einer dafür geeigneten Kommunikationsform.

## *190. Shed Load*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 201-202

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

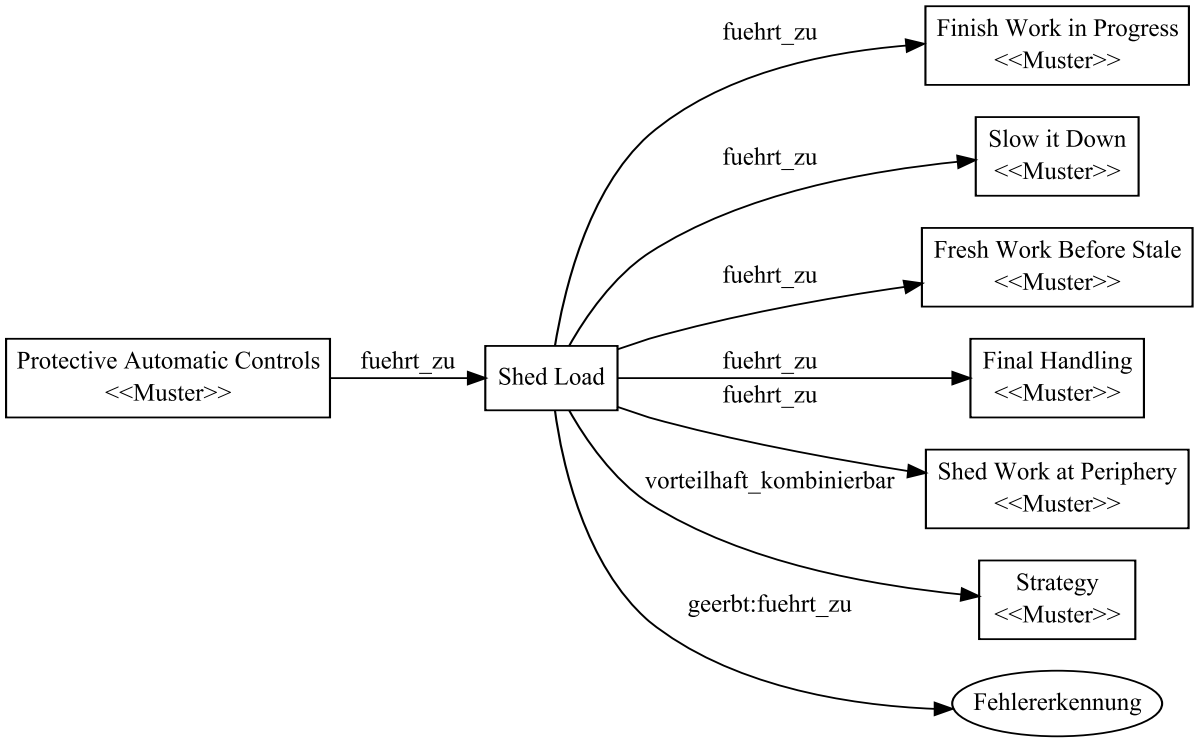
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Shed Load" beschreibt den Ansatz, in Überlast-Situationen Anfragen zu verwerfen, sodass die restlichen Anfragen in akzeptabler Zeit bearbeitet werden können.

Beziehungen - Grafik




Beziehungen - Detail

- **Protective Automatic Controls -> Shed Load** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Shed Load -> Finish Work in Progress** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 202
  *"Finish work in progress discusses a way to intelligently select the work requests that should be rejected."*
- **Shed Load -> Slow it Down** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Shed Load -> Fresh Work Before Stale** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Shed Load -> Final Handling** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Shed Load -> Shed Work at Periphery** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Shed Load -> Strategy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 202
  *"How to handle work that arrives from people depends on the circumstances. For example telephone calls and web clicks have different human behavior characteristics. The Strategy pattern is useful for encoding the needed variation between these scenarios."*
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung

Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *191. Shed Work at Periphery*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 208-209

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

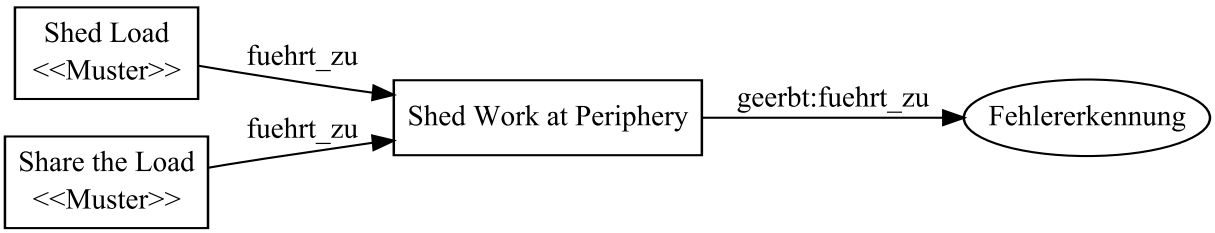
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Shed Work at Periphery" beschreibt die Steigerung zum Pattern "Shed Load". Treffen mehr Anfragen ein, als das System selbst verwerfen kann, so ist es erforderlich, Anfragen bereits zu verwerfen oder zu blockieren bevor sie das eigentliche System erreichen.

Beziehungen - Grafik




Beziehungen - Detail

- **Shed Load -> Shed Work at Periphery** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Share the Load -> Shed Work at Periphery** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *192. Single Access Point*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 279-286

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

**Paper: Architectural Patterns for Enabling Application Security, Yoder**

S. 4

Joseph Yoder und Jeffrey Barcalow, “Architectural Patterns for Enabling Application Security”, 1998, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274.

Schwerpunkt(e)

- Sicherheit

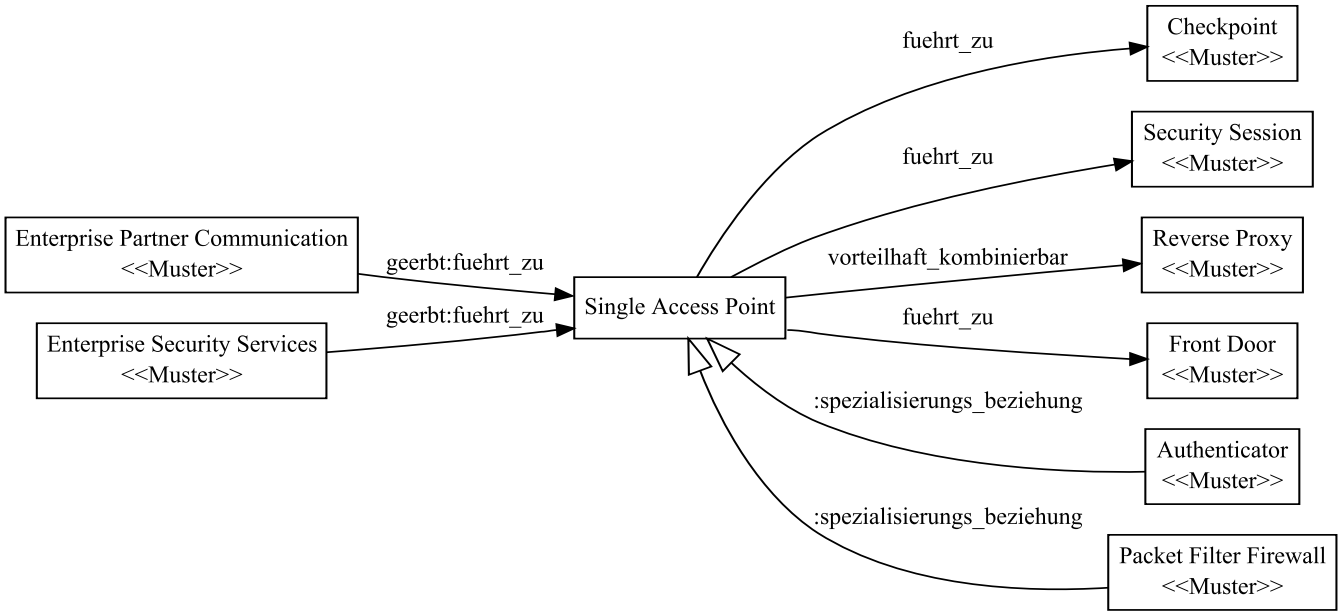
Gruppe(n)

- Point of Access

Kurzbeschreibung

Das Pattern "Single Access Point" beschreibt den Ansatz, den Zugang zu einem System über ausschließlich einen Weg zu erlauben, sodass auch nur dieser eine Zugriffsweg geschützt werden muss.

Beziehungen - Grafik



Beziehungen - Detail

- **Single Access Point -> Checkpoint** : Zugangskontrolle
  Quelle: Security Patterns; Schumacher et al; S. 284
  *“Optionally implement the entry check at the single access point.* [...] *Checkpoint shows how to make this checking flexible.“*
- **Single Access Point -> Security Session** : day pass
  Quelle: Security Patterns; Schumacher et al; S. 284
  Zweck: Erhalt der beim Checkpoint ausgestellten Identitäts- und Berechtigungsinformation
- **Single Access Point -> Reverse Proxy** :
  Ein Single Access Point kann durch eine Ausprägung von Reverse Proxy implementiert werden.
  Quelle: Security Patterns; Schumacher et al.; S. 286
  *“Other patterns in this book, such as the firewall patterns in Chapter 12 and Protection Reverse Proxy (457), provide examples of effective single access points, in which the clients are not always users, but can be network traffic that needs entry to the protected system.”*
- **Single Access Point -> Front Door** : Agiert als
  Der Integration Reverse Proxy des Front-Door-Patterns agiert als einziger Zugriffsweg (Single Access Point) zu den dahinter liegenden Systemen.
  Quelle: Security Patterns; Schumacher et al; S. 479
  *“You can view Front Door as adding Checkpoint and Security Session to an Integration Reverse Proxy or Protection Reverse Proxy, and thus also providing a Single Access Point to a company´s Web applications and services.”*
- **Single Access Point -> Authenticator** : Spezialform
  Quelle: Security Patterns, Schumacher et al; S. 327
  *"Single access point is an abstract pattern applied here: Authenticator is a concrete application of it."*
- **Single Access Point -> Packet Filter Firewall** :
  Quelle: Security Patterns; Schumacher et al.; S. 410;
  *“This pattern is also a special case of Single Access Point* [...]*”*
- **Enterprise Partner Communication -> Point of Access** : Gestaltung des Zugriffspunkts für Partner
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Enterprise-Partner-Communication-Pattern beschreibt die Kommunkationsanforderungen zwischen Organisationen mit voneinander getrennt existierender IT-Infrastruktur. Zur Umsetzung der Enterprise Partner Communication muss ein Muster für die Gestaltung des Point of Access ausgewählt werden, um für den Kommunikationspartner einen Zugangsweg zu dem betreffenden IT-System zu eröffnen.
- **Enterprise Security Services -> Point of Access** : Gestaltung eines sicheren Zugriffspunkts für interne Applikationen
  Geerbte Beziehung: fuehrt_zu_beziehung

Um einen sicheren Zugriffsweg auf interne Applikationen zu gestalten sollte ein Pattern der Gruppe Point of Access angewendet werden.

## *193. Single Storage-Point per Patient*

Schwerpunkt(e)

- Kommunikation

Gruppe(n)

- Verteilung der Dokumentenablage pro Patient

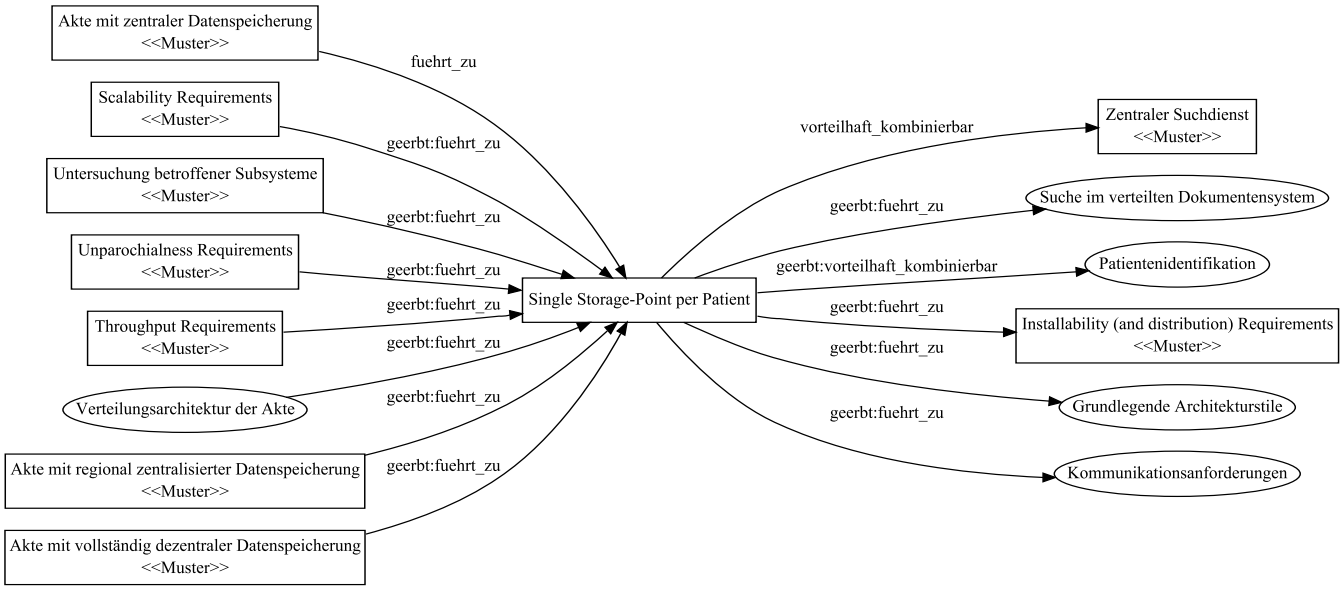
Kontext
Die Anforderungen an Zuverlässigkeit, Skalierbarkeit und andere Kriterien sind definiert. Die technische Umgebung des Systems wurde untersucht und dokumentiert. Die Ergebnisse dieser Untersuchung zeigen nicht eindeutig, dass eine umfassende Kompetenz zum eigenständigen Betrieb von IT-Systemen bei allen beteiligten Einrichtungen vorliegt. Es besteht kein Zwang, Dokumente ausschließlich auf IT-Systemen in der Einrichtung ihrer Entstehung zu speichern.

Problem
Wie kann man Dokumente zu einem Patienten so ablegen, dass die Auffindung der einzelnen Dokumente möglichst einfach realisierbar ist. Das beinhaltet sowohl eine möglichst einfache Verwaltung der Ablageorte als auch eine möglichst einfache und wenig laufzeitintensive Suche nach Dokumenteninhalten.

Lösung
Die Lösung des Problems ist die Speicherung aller Dokumente zu einem Patienten auf einem Systemknoten in einer dort vorliegenden zusammenhängenden Akte. Dadurch ist nur die Identifikation des Systemknotens, der zur Ablage der Dokumente des Patienten zu verwenden ist, im verteilten System notwendig. Alle weiteren Operationen können auf diesem System lokal ausgeführt werden, ohne dass die Operationen selbst zur Erfüllung ihrer Aufgaben auf weitere entfernte Kommunikation angewiesen sind.

Das Pattern Single Storage-Point per Patient kann mit allen Patterns der Gruppe Verteilungsarchitektur der Akte kombiniert werden.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Akte mit zentraler Datenspeicherung -> Single Storage-Point per Patient** : kann nur als realisiert werden
  Wenn nur ein einziger (zentraler) Ort zur Datenspeicherung im gesamten Aktensystem existiert, können die Daten zu einem Patienten auch nur an diesem einen Ort gespeichert sein.
- **Single Storage-Point per Patient -> Zentraler Suchdienst** :
  Die Verwendung des Single-Storage-Point-per-Patient-Patterns vereinfacht den Aufbau eines zentralen Suchdienstes. Durch die eindeutige Zuordnung zwischen Patient und speicherndem Systemknoten kann dieser bereits zu Beginn einer Suchanfrage eindeutig identifiziert werden. Das ermöglicht ein eindeutiges Routing der Suchanfrage an einen jeweils einzelnen zuständigen Systemknoten. Dadurch wird eine zum Teil unnötige Belastung der anderen Systemknoten bei Suchanfragen vermieden. Ein weiterer Vorteil dieser Kombination liegt darin, dass im zentralen Suchdienst lediglich ein Index über die Zuordnung zwischen Knoten und Patient und nicht über alle verfügbaren Dokumente aufgebaut und vorgehalten werden muss.
- **Scalability Requirements -> Verteilung der Dokumentenablage pro Patient** :
  Skalierbarkeitsanforderungen beeinflussen Verteilung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Patterns der Gruppe Verteilung der Dokumentenablage pro Patient beschreiben Architekturvarianten für die Gestaltung der Ablage von Dokumenten. Die zentrale Aufgabe einer verteilten Krankenakte beinhaltet den regelmäßigen schreibenden und lesenden Zugriff auf Dokumente zu Patienten. Die Skalierbarkeitsanforderungen beschreiben, in welcher Form und welchem Umfang das System wachsen und schrumpfen können soll. Diese Fähigkeit basiert unter anderem auf der Architektur der Dokumentenablage. Deshalb wird diese durch die Skalierbarkeitsanforderungen direkt beeinflusst.

- **Untersuchung betroffener Subsysteme -> Verteilung der Dokumentenablage pro Patient** : Verwendbarkeit der Subsysteme
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Unparochialness Requirements -> Verteilung der Dokumentenablage pro Patient** : schränken ein
  Geerbte Beziehung: fuehrt_zu_beziehung
  Gewählte Unparochialness Requirements schränken die Menge der möglichen Verteilungslösungen ein. Abgeleitet aus (Withall, S. 260 Punkt 5, External Systems) -> Anforderungen an Kommunikationsfähigkeiten zu externen Systemen.
- **Throughput Requirements -> Verteilung der Dokumentenablage pro Patient** : Beeinflusst die Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Beispiel:
  Werden in einer verteilten Krankenakte Dokumente hauptsächlich in der Einrichtung ihrer Erstellung benutzt, so wird die maximale Kapazität für den Durchsatz durch vollständig dezentral gespeicherte Dokumente (Fully Distributed Patient Documents) erreicht. Dabei wird schon bei der Erstellung des Dokuments durch ortsnahe Speicherung und die daraus resultierende Nutzung lokaler Hochgeschwindigkeitsnetze bei der gleichzeitig keine Belastung für dritte Systeme entsteht, hoher Datendurchsatz ermöglicht. Da dritte Systeme nicht belastet werden, steht deren volle Leistungsfähigkeit ebenfalls für den lokalen Nutzungsfall zur Verfügung.
  Verändert sich die Art der Dokumentennutzung allerdings hin zu einer verstärkten Nutzung von Dokumenten anderer Einrichtungen, so sinkt der mögliche Durchsatz deutlich, da Suchvorgänge über eine Vielzahl dezentraler Knoten notwendig werden. Diese Vorgänge belasten gleichzeitig viele Systemknoten, die dann für den lokalen Anwendungsfall nicht mehr ihre volle Leistungsfähigkeit zur Verfügung stellen können. In diesem Szenario entwickelt das andere Pattern der Gruppe Verteilung der Dokumentenablage pro Patient seine Vorteile, da hier auch für starke verteilte Nutzung pro Patient immer nur ein Systemknoten für Suchvorgänge belastet wird.
- **Verteilungsarchitektur der Akte -> Verteilung der Dokumentenablage pro Patient** : unterschiedlich kombinierbar
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Akte mit regional zentralisierter Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Verwendung einer Akte mit regional zentralisierter Datenspeicherung muss festgelegt werden, ob die Daten zu einem Patienten in einem, dem Patienten fest zugeordneten regionalen Knoten gespeichert werden oder in pro Patient möglicherweise mehreren Knoten abgelegt werden sollen. Der letztere Fall kann z. B. auftreten, wenn der Ablageort nicht dem Patienten regional zugeordnet, sondern der Einrichtung, die den Patienten behandelt regional zugeordnet wird.
- **Akte mit vollständig dezentraler Datenspeicherung -> Verteilung der Dokumentenablage pro Patient** : unterstützt beide Ausprägungen
  Geerbte Beziehung: fuehrt_zu_beziehung

- **Verteilung der Dokumentenablage pro Patient -> Suche im verteilten Dokumentensystem** : Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Wenn Dokumente verteilt abgelegt werden, ist es notwendig sie wiederfinden zu können.
- **Verteilung der Dokumentenablage pro Patient -> Patientenidentifikation** :
  Geerbte Beziehung: vorteilhaft_kombinierbar_beziehung
  Wenn Dokumente verteilt abgelegt werden ist es hilfreich, wenn die zugehörigen Patienten über die verschiedenen Systemknoten hinweg eindeutig identifizierbar sind.
- **Verteilung der Dokumentenablage pro Patient -> Installability (and distribution) Requirements** : Anforderungen zur verteilten Installation
  Geerbte Beziehung: fuehrt_zu_beziehung
  Je stärker die Dokumentenablage verteilt werden soll, desto wichtiger wird es, dass die Softwaresysteme der Knoten auf denen die Dokumentenablage erfolgt, einfach zu installieren ist.
- **Verteilung der Dokumentenablage pro Patient -> Grundlegende Architekturstile** : Aus Verteilung resultieren zusätzliche Anforderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Abhängig davon, ob und wie die Dokumente verteilt abgelegt werden, resultieren daraus Einschränkungen für die Auswahl grundlegender Architekturstile.
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.

## *194. Singleton*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 127-134

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

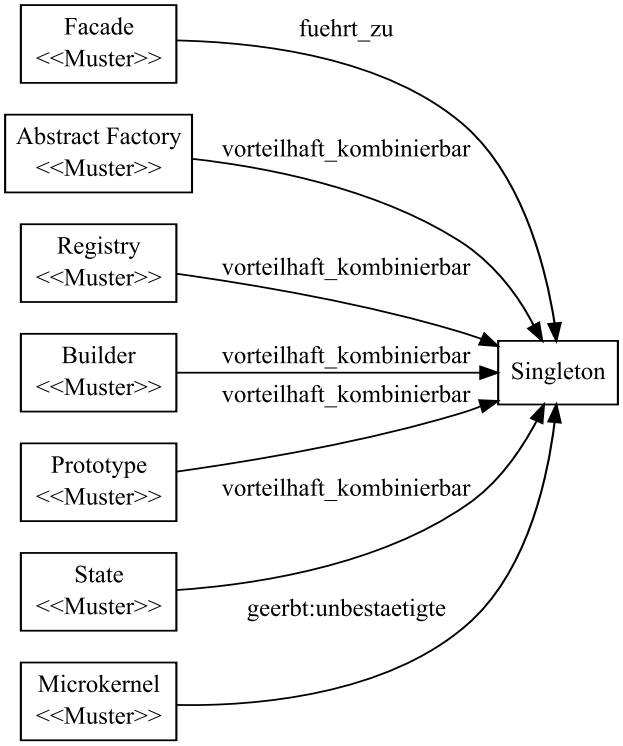
Gruppe(n)

- Instanziierung und Kontrolle der Instanzen

## Kurzbeschreibung

Das Pattern "Singleton" beschreibt einen erprobten Weg, um sicherzustellen, dass von einer Klasse innerhalb eines Prozesses nur genau eine Instanz erzeugt wird.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Facade -> Singleton** : Eine Instanz
  Um von einer Facade nur eine Instanz zu haben wird ein Singleton benötigt. (Siehe Gamma letzte Seite, Design Pattern Relationships).
  Quelle: Design Patterns, Gamma et al.; S. 193
  *“Usually only one Facade object is required. Thus Facade objects are often Singletons.“*
- **Abstract Factory -> Singleton** : for single instance
  Um von einer abstrakten Factory nur eine Instanz zu generieren und diese überall verwenden zu können, wird die Kombination mit dem Singleton Muster benötigt.
  Quellen:
  - Siehe letzte Seite, Design Pattern Relationships
  - und S. 95 unten *"AbstractFactory classes are often implemented with factory methods, but they can also be implemented using Prototype. A concrete factory is often a singleton."*

  in Gamma, Johnson, Vlissides, Design Patterns

- **Registry -> Singleton** : Häufig kombiniert
  Quelle: Patterns of Enterprise Application Architecture, Fowler, S. 481 (3. Absatz)
  *"For a process-scoped Registry, then, the usual option is a singleton."*
- **Builder -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 134
  *"Many patterns can be implemented using the Singleton pattern. See Abstract Factory (87), Builder(97), and Prototype (117)."*
- **Prototype -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 134
  *"Many patterns can be implemented using the Singleton pattern. See Abstract Factory (87), Builder(97), and Prototype (117)."*
- **State -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 313
  *"State objects are often Singletons"*
- **Microkernel -> Instanziierung und Kontrolle der Instanzen** : zur Instanziierung der Erweiterungen
  Geerbte Beziehung: unbestaetigte_beziehung
  Diese Beziehung existiert, da bei einer Microkernel-Architektur die Erweiterungen auch instanziiert und eingebunden werden müssen. Die Patterns der referenzierten Gruppe (Factory und Singleton) können zur Instanzierung der Plug-Ins selbst oder ihrer Proxys verwendet werden.
  Basis dieser Überlegungen sind die Seiten 173 - 192 in Pattern oriented Software Architecture, Volume 1.

## *195. Slow it Down*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 210-213

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

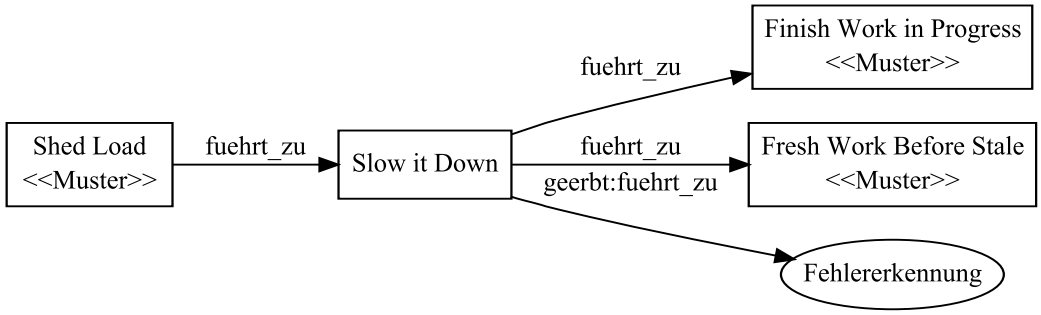
Gruppe(n)

- Fehlerkompensation

Kurzbeschreibung

Das Pattern "Slow it Down" beschreibt ein auf Eskalationsstufen basierendes Konzept, die eigentlichen Verarbeitungsprozesse auf hoher Priorität laufen zu lassen, während verschiedene andere Prozesse - besonders solche die neue Anfragen annehmen - auf niedrigerer Priorität (und somit verlangsamt) betrieben werden.

Beziehungen - Grafik




Beziehungen - Detail

- **Shed Load -> Slow it Down** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Slow it Down -> Finish Work in Progress** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 182
- **Slow it Down -> Fresh Work Before Stale** :
  Quelle: Patterns for Fault Tolerant Software, Hanmer, S. 182
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *196. Small Patches*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 239-241

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

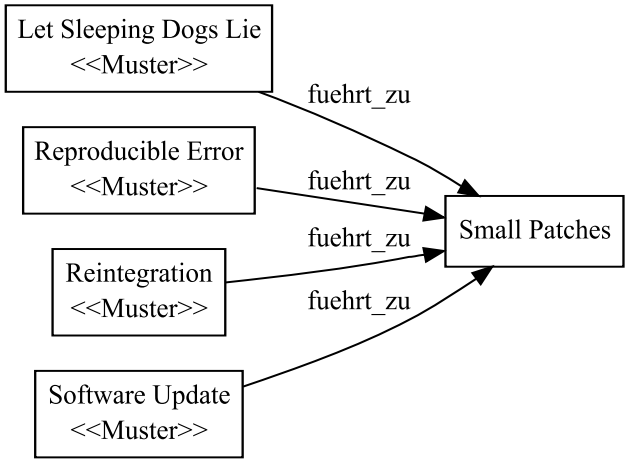
Gruppe(n)

- Fehlerkorrektur

Kurzbeschreibung

Das Pattern "Small Patches" beschreibt den Ansatz, Fehler besser durch individuelle - also eher kleinere und gut testbare - Patches zu beheben.

Beziehungen - Grafik




Beziehungen - Detail

- **Let Sleeping Dogs Lie -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Reproducible Error -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Reintegration -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 228
- **Software Update -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *197. Software Update*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 78-81

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

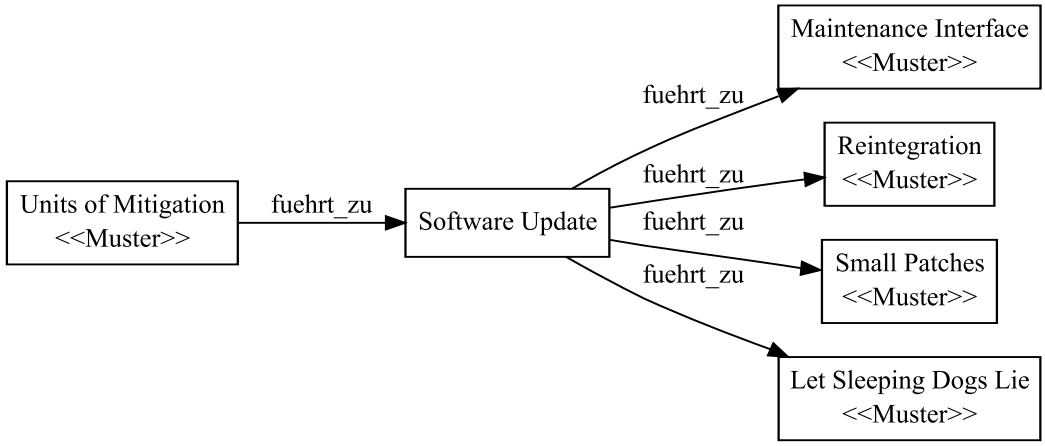
- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Software Update" beschreibt die Notwendigkeit der Berücksichtigung von Update-Mechanismen bereits bei der Systemkonzeption. Das gilt besonders dann, wenn ein

System während des Updates verfügbar bleiben muss oder die Dauer der Update-bedingten Downtime nur minimal sein darf.

Beziehungen - Grafik




Beziehungen - Detail

- **Units of Mitigation -> Software Update** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Software Update -> Maintenance Interface** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Software Update -> Reintegration** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Software Update -> Small Patches** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Software Update -> Let Sleeping Dogs Lie** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *198. Someone in Charge*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 66-68

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

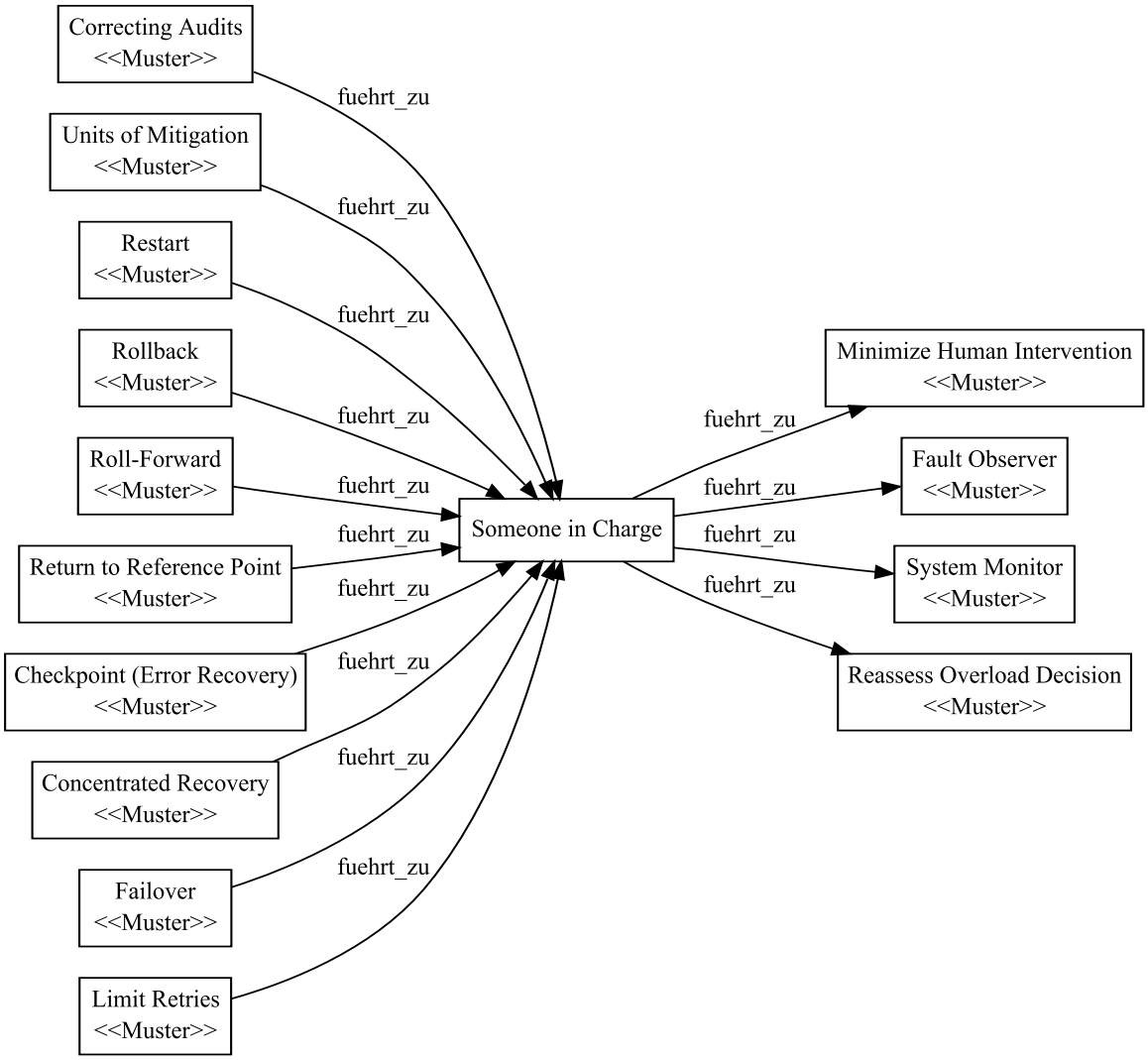
## Gruppe(n)

- Architekturpatterns Fehlertoleranz

## Kurzbeschreibung

Das Pattern "Someone in Charge" befasst sich mit der Frage der Abdeckung bei der Behandlung von Fehlern und der Behandlung von Fehlern im Code der Fehlerbehandlung.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Correcting Audits -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Units of Mitigation -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Restart -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Rollback -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Roll-Forward -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Return to Reference Point -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Checkpoint (Error recovery) -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Concentrated Recovery -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Failover -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Limit Retries -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; Letzte Seite, A Pattern Language for Fault Tolerant Software;
- **Someone in Charge -> Minimize Human Intervention** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Someone in Charge -> Fault Observer** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Someone in Charge -> System Monitor** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Someone in Charge -> Reassess Overload Decision** :
  Quelle: Patterns for Fault Tolerant Systems; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

## *199. State*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 305-313

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

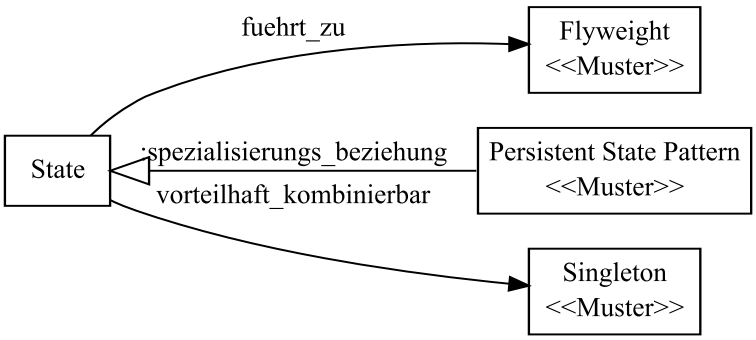
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das State-Pattern beschreibt einen Ansatz, der ermöglicht das Verhalten der Methoden abhängig vom Zustand des Objekts zu variieren.

Beziehungen - Grafik




Beziehungen - Detail

- **State -> Flyweight** : sharing states
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma et al.; S. 206;
  *"It's often best to implement State and Strategy objects as flywights"*
- **State -> Persistent State Pattern** :
  vgl. http://www.hillside.net/plop/2010/papers/saude.pdf
- **State -> Singleton** : Umsetzbar als
  Quelle: Design Patterns; Gamma et al; S. 313
  *"State objects are often Singletons."*

## *200. Stateful Firewall*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 417-421

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

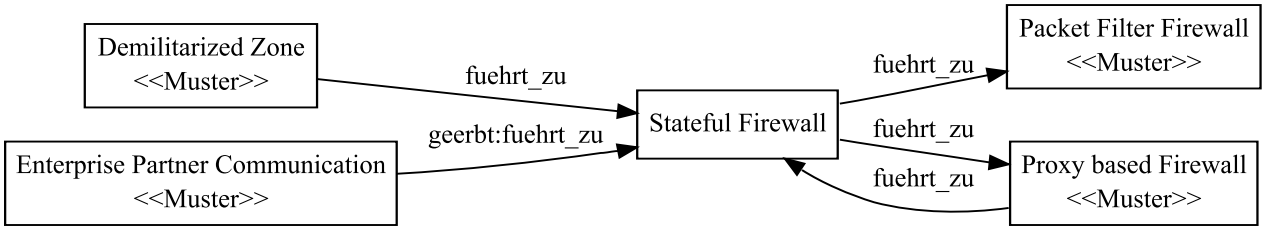
Gruppe(n)

- Access Restriction

Kurzbeschreibung

Das Pattern "Stateful Firewall" beschreibt einen Firewall-Typ, der den Zustand der betrachteten Verbindungen berücksichtigt und nicht nur auf Ebene der einzelen Paketinhalte filtert.

Beziehungen - Grafik




Beziehungen - Detail

- **Demilitarized Zone -> Stateful Firewall** : Bestandteil
  Zur Umsetzung einer DMZ ist eine Firewall notwendig.
  Quelle: S. 451 ff.; Security Patterns; Schumacher et al.
- **Proxy based Firewall -> Stateful Firewall** : Zum Schutz gegen zusätzliche Angriffsformen
  Quelle: Security Patterns; Schumacher et al.; S.416
  *„This pattern uses the Proxy pattern from GoF. It can be combined with Packet Filter Firewall and Stateful Firewall"*
  Quelle: Security Patterns; Schumacher et al.; S.417
  *"We have been able to contain many attachs with Packet Filter Firewall and Proxy-Based Firewall. However, we are still plagued with distributed denial of service attacks that prevent customers from reaching our site. We also have* [...]*"*
- **Stateful Firewall -> Packet Filter Firewall** : Üblicherweise kombiniert
  Quelle: Security Patterns; Schumacher et al; S. 421
  *"This firewall* [Stateful Firewall] *is usually combined with one or both of the previous types of firewalls, Packet Filter Firewall and Proxy-Based Firewall"*
- **Stateful Firewall -> Proxy based Firewall** : Üblicherweise kombiniert
  Quelle: Security Patterns; Schumacher et al; S. 421
  *"This firewall* [Stateful Firewall] *is usually combined with one or both of the previous types of firewalls, Packet Filter Firewall and Proxy-Based Firewall"*
- **Enterprise Partner Communication -> Access Restriction** : EPC als Rahmenbedingungen für AR
  Geerbte Beziehung: fuehrt_zu_beziehung
  Access Restriction beinhaltet Mechanismen wie DMZs, Firewalls usw. Diese finden vor allem im Rahmen der Enterprise Partner Communication Anwendung.

## *201. Static Capacity Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 215-217

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Zuverlässigkeit

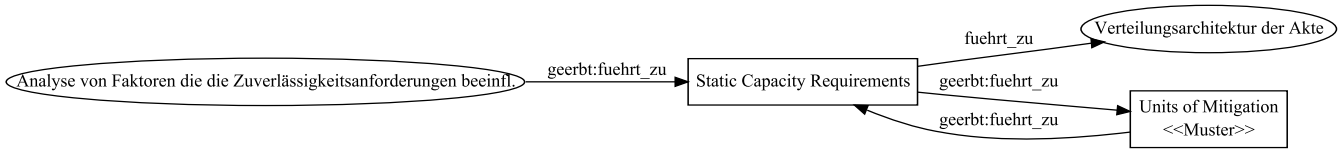
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Static Capacity Requirements beschreiben die Anforderungen, die an die Menge an Daten, Datensätzen usw. die ein System aufnehmen können soll, gestellt werden.

Beziehungen - Grafik




Beziehungen - Detail

- **Static Capacity Requirements -> Verteilungsarchitektur der Akte** : Beeinflusst die Auswahl
  Static Capacity Requirements beschreiben die Menge an Daten bzw. Dokumenten, die die verteilte Krankenakte oder einer ihrer Bestandteile zu speichern in der Lage sein muss. Die Menge dieser Daten kann für die Entscheidung zugunsten einer zentralen oder dezentralen Architektur maßgeblich sein.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.

- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.
- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## *202. Strategy*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 315-323

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

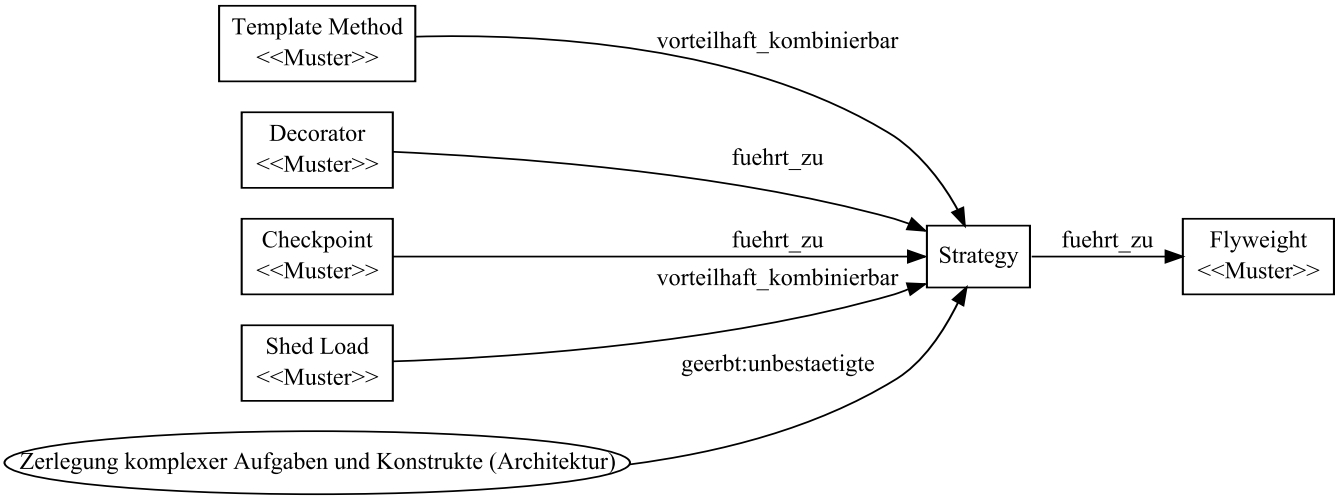
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)

Kurzbeschreibung

Das Strategy-Pattern beschreibt einen Ansatz, verschiedene Algorithmen für das gleiche Problem leicht austauschbar im verwendenden Code zu benutzen.

Beziehungen - Grafik




Beziehungen - Detail

- **Template Method -> Strategy** : defining algorithms steps
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Decorator -> Strategy** : changing skin versus guts
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Checkpoint -> Strategy** : Erhöhung der Flexibilität
  Quelle: Security Patterns, Schumacher et al., S. 296;
  *"Check Point uses Strategy for gaining flexibility in application security"*
- **Shed Load -> Strategy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 202
  *"How to handle work that arrives from people depends on the circumstances. For example telephone calls and web clicks have different human behavior characteristics. The Strategy pattern is useful for encoding the needed variation between these scenarios."*
- **Strategy -> Flyweight** : sharing strategies
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma et al.; S. 206;
  *"It's often best to implement State and Strategy objects as flywights"*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *203. System Monitor*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 98-100

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

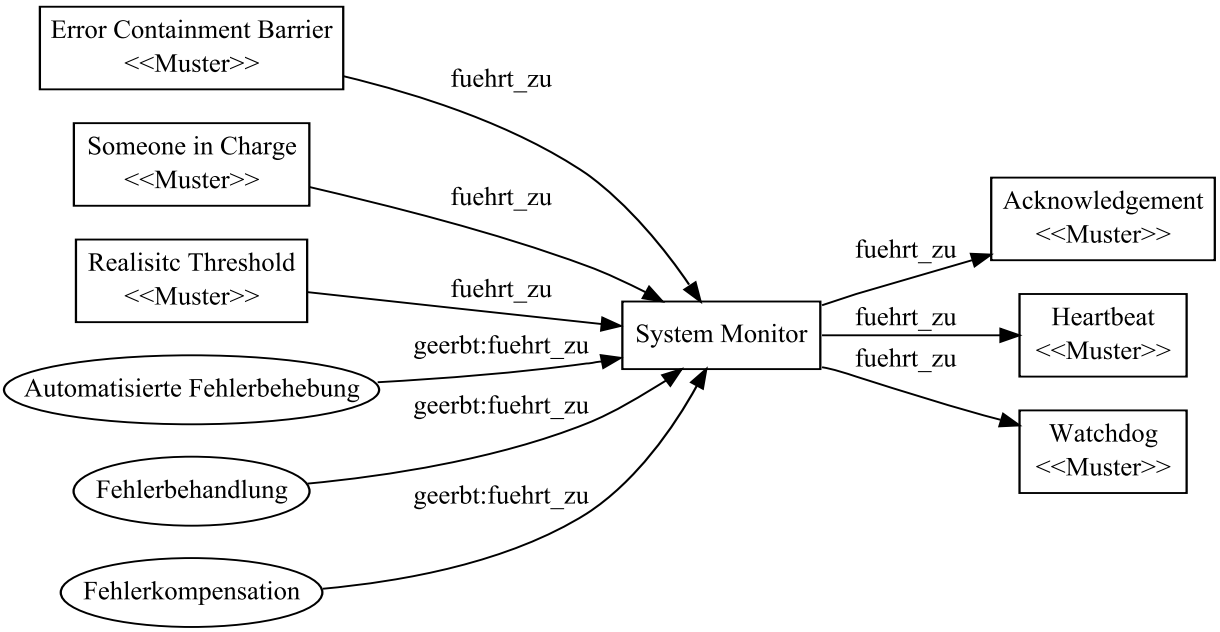
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "System Monitor" beschreibt einen Ansatz zur Überwachung der Funktionsfähigkeit von einzelnen, mehreren oder allen Komponenten eines Softwaresystems.

Beziehungen - Grafik




Beziehungen - Detail

- **Error Containment Barrier -> System Monitor** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Someone in Charge -> System Monitor** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software

- **Realisitc Threshold -> System Monitor** : Wenn Fehler erkannt wurde
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 114
  *"Refer to System Monitor for a discussion of what steps it can take when the error is detected."*
- **System Monitor -> Acknowledgement** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **System Monitor -> Heartbeat** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **System Monitor -> Watchdog** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *204. Technology Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 65-70

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Kommunikation

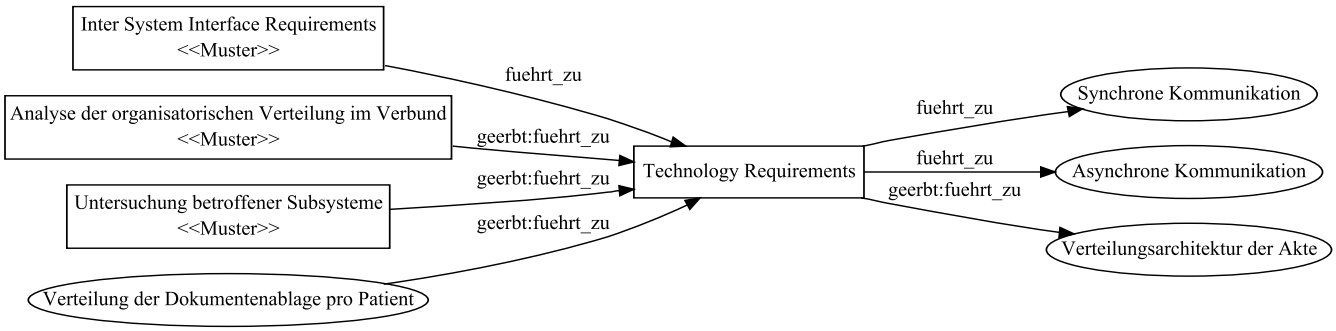
Gruppe(n)

- Kommunikationsanforderungen

## Kurzbeschreibung

Das Pattern "Technology Requirements" dient der Spezifikation von Anforderungen bezüglich der Technologien, die aufgrund bestehender Rahmenbedingungen genutzt werden müssen.

## Beziehungen - Grafik




## Beziehungen - Detail

- **Inter System Interface Requirements -> Technology Requirements** : Spezifikation der verwendeten Technologie
  Das Inter-System-Interface-Requirements-Pattern dient der Definition von Schnittstellen in Form von Anforderungen. Die beschriebenen Schnittstellen können durch Hinzufügen von Technology Requirements um die Spezifikation der zur Umsetzung zu verwendenden Technologien erweitert werden.
  Quelle: Software Requirement Patterns;Withall; S. 55
  *"7. Technology to be used for the interface (if relevant)* [...]*"*
- **Technology Requirements -> Synchrone Kommunikation** : Auswahl Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Technology Requirements -> Asynchrone Kommunikation** : Auswahl Technology Requirements können durch die Festlegung auf eine für das gesamte Projekt oder seine Teile zu verwendende Kommunikationstechnologie frühzeitig die Auswahl zwischen synchroner und asynchroner Kommunikation im Architekturkonzept festschreiben. Eine implizite Auswahl der Kommunikationsform liegt immer dann vor, wenn ein Kommunikationsframework ausgewählt wird, das nicht beide Kommunikationsformen unterstützt.
- **Analyse der organisatorischen Verteilung im Verbund -> Kommunikationsanforderungen** : Input: Menge der Übertragungswege
  Geerbte Beziehung: fuehrt_zu_beziehung
  Im Rahmen der Anwendung des Musters "Analyse der organisatorischen Verteilung im Verbund" werden die an der verteilten Krankenakte beteiligten Einrichtungen ermittelt. Aus der Menge der Einrichtungen wiederum resultiert die maximale Menge

der Kommunikanten, zwischen denen Kommunikationswege aufgebaut und in abgesicherter Form zur Verfügung gestellt werden müssen.

- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** :
  Input: Datenquellen, Schnittstellen und Standards
  Geerbte Beziehung: fuehrt_zu_beziehung
- **Verteilung der Dokumentenablage pro Patient -> Kommunikationsanforderungen** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das gewählte Prinzip nach dem die Dokumente eines Patienten gespeichert bzw. verteilt gespeichert werden, beeinflusst direkt, wie innerhalb des Gesamtsystems kommuniziert werden kann bzw. muss.
- **Kommunikationsanforderungen -> Verteilungsarchitektur der Akte** : Auswahl
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die ermittelten Kommunikationsanforderungen werden zur Auswahl einer geeigneten Verteilungsarchitektur der Akte verwendet.

## *205. Template Method*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 325-330

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

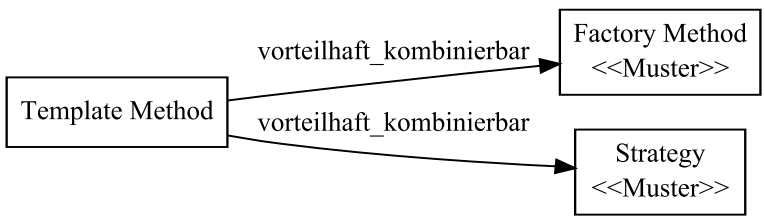
Gruppe(n)

- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Pattern "Template Method" beschreibt einen Weg, für alle Bestandteile einer Vererbungshierarchie invariante Bestandteile von Methoden einmalig an zentraler Stelle und in allgemeingültiger Form umzusetzen.

Beziehungen - Grafik



Beziehungen - Detail

- **Template Method -> Factory Method** : Benutzt häufig
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Template Method -> Strategy** : defining algorithms steps
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships

## *206. Threat Assessment*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 113-124

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

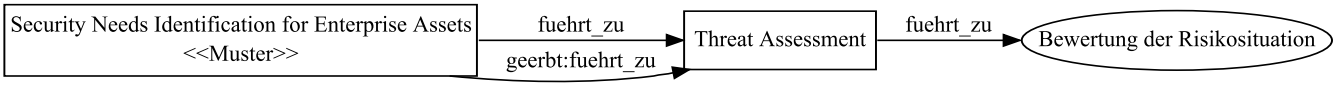
Gruppe(n)

- Analyse der Sicherheitssituation

Kurzbeschreibung

Das Pattern "Threat Assessment" beschreibt ein Verfahren zur Bewertung von Bedrohungen. Ergebnis ist eine tabellarische Darstellung der bewerteten Bedrohungen.

Beziehungen - Grafik




Beziehungen - Detail

- **Security Needs Identification for Enterprise Assets -> Threat Assessment** :
  Analyse der Bedrohungssituation
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Threat Assessment -> Bewertung der Risikosituation** : Input: Menge der identifizierten Bedrohungen
  Quelle: Security Patterns; Schumacher et al; S. 60 u. 61
  Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
  Quelle: Security Patterns; Schumacher et al; S. 147;Risk Determination, Liabilities

*"The results are based on the completeness and subjectivity of Asset Valuation, Threat Assessment and Vulnerability Assessment* [...]*."*

- **Security Needs Identification for Enterprise Assets -> Analyse der Sicherheitssituation** : Grobanalyse zu Detailanalyse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Security-Needs-Identification-for-Enterprise-Assets-Pattern dient der Identifikation und Bewertung gefährdeter Güter, Anlagen, Personen und Daten in den untersuchten medizinischen Versorgungseinrichtungen. Die ermittelten Gefährdeten, sowie das ebenfalls ermittelte Einflusspotenzial auf den Gesamtbetrieb dienen dazu im Rahmen der Analyse der Sicherheitssituation ermittelte Sicherheitsrisiken und andere Einflussfaktoren nach ihrer Wichtigkeit zu bewerten.

## *207. Throughput Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 204-211

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Zuverlässigkeit

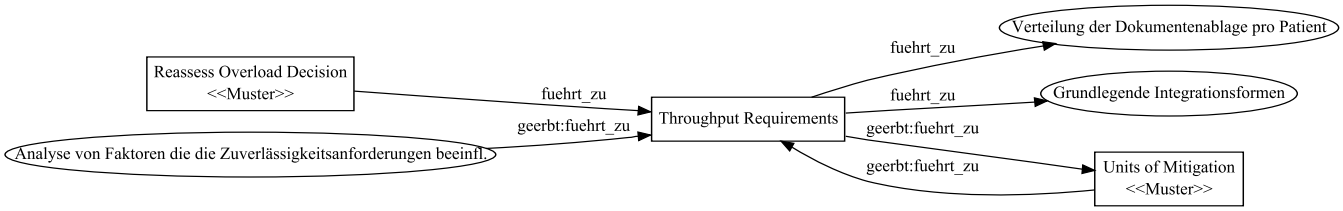
Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen

Kurzbeschreibung

Das Pattern "Throughput Requirements" beschreibt die Definition von Anforderungen bezüglich des Durchsatzes, den das System bezogen auf ein zu definierendes Kriterium (Anzahl einer bestimmten Art von Anfragen, Menge übertragbarer Daten usw.) mindestens bewältigen können muss.

Beziehungen - Grafik




Beziehungen - Detail

- **Reassess Overload Decision -> Throughput Requirements** : Anpassen der Anforderungen.

Das Reassess-Overload-Decision-Pattern behandelt die Anpassung der Auswahl von Mitteln zur Behandlung von durch Überlastung des Systems eingetretenen Problemen. Das Eintreffen einer großen Zahl von Anfragen, Nachrichten oder allgemein Daten ist ein typischer Grund für hohe Systembelastung. Die Menge an Anfragen, Nachrichten oder Daten, die ein System zu einem Zeitpunkt bewältigt, wird als Durchsatz bezeichnet. Verändert sich der Umfang zu erwartender Überlastszenarien so sind folglich die Anforderungen an den Durchsatz (Throughput Requirements) anzupassen.

- **Throughput Requirements -> Verteilung der Dokumentenablage pro Patient** : Beeinflusst die Auswahl
  Beispiel:
  Werden in einer verteilten Krankenakte Dokumente hauptsächlich in der Einrichtung ihrer Erstellung benutzt, so wird die maximale Kapazität für den Durchsatz durch vollständig dezentral gespeicherte Dokumente (Fully Distributed Patient Documents) erreicht. Dabei wird schon bei der Erstellung des Dokuments durch ortsnahe Speicherung und die daraus resultierende Nutzung lokaler Hochgeschwindigkeitsnetze bei der gleichzeitig keine Belastung für dritte Systeme entsteht, hoher Datendurchsatz ermöglicht. Da dritte Systeme nicht belastet werden, steht deren volle Leistungsfähigkeit ebenfalls für den lokalen Nutzungsfall zur Verfügung.
  Verändert sich die Art der Dokumentennutzung allerdings hin zu einer verstärkten Nutzung von Dokumenten anderer Einrichtungen, so sinkt der mögliche Durchsatz deutlich, da Suchvorgänge über eine Vielzahl dezentraler Knoten notwendig werden. Diese Vorgänge belasten gleichzeitig viele Systemknoten, die dann für den lokalen Anwendungsfall nicht mehr ihre volle Leistungsfähigkeit zur Verfügung stellen können. In diesem Szenario entwickelt das andere Pattern der Gruppe Verteilung der Dokumentenablage pro Patient seine Vorteile, da hier auch für starke verteilte Nutzung pro Patient immer nur ein Systemknoten für Suchvorgänge belastet wird.
- **Throughput Requirements -> Grundlegende Integrationsformen** : beeinflusst die Auswahl
  Die verschiedenen Integrationsformen besitzen unterschiedliche Eigenschaften bezüglich ihrer Fähigkeit, große Mengen von Anfragen oder große Mengen von Daten in wenigen Anfragen zu verarbeiten.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung

Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.

- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## *208. Units of Mitigation*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 37-41

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

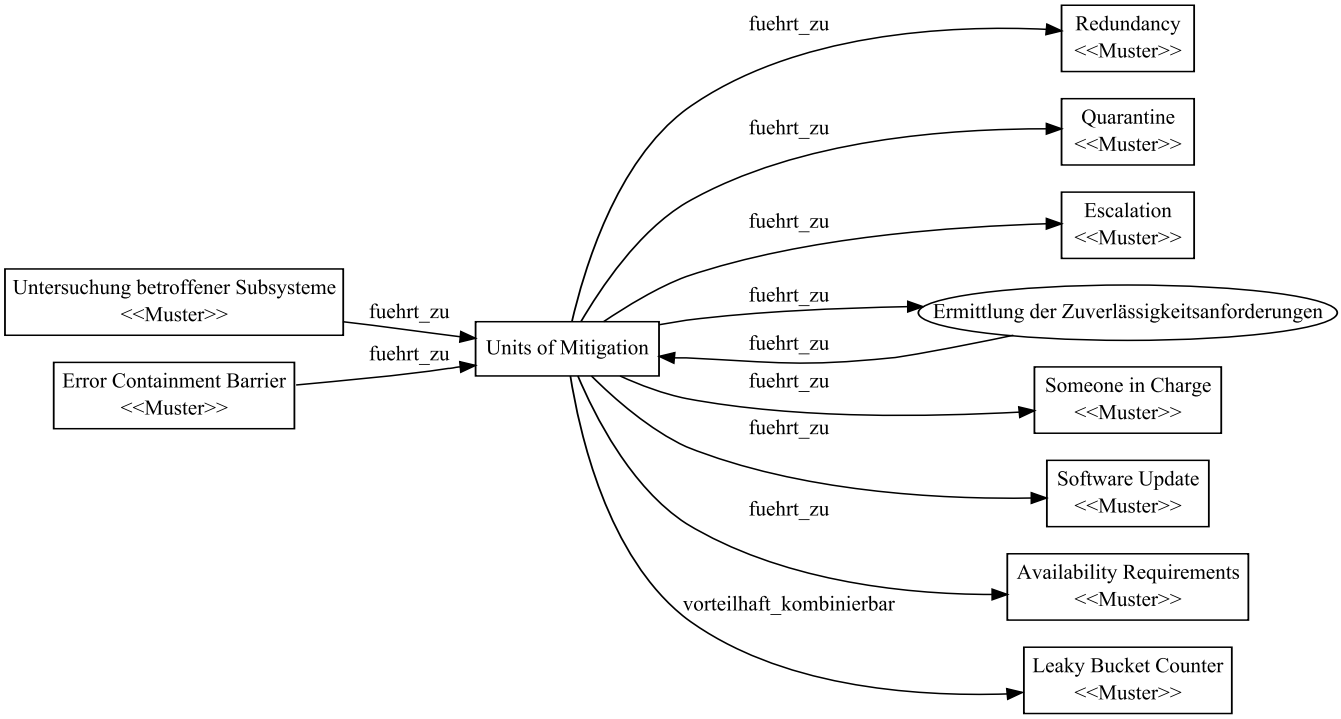
Gruppe(n)

- Architekturpatterns Fehlertoleranz

Kurzbeschreibung

Das Pattern "Units of Mitigation" beschreibt den Ansatz, ein System in eine Menge logischer Einheiten zu gliedern, die bezüglich der Fehlerbehandlung so von den anderen logischen Einheiten abgegrenzt sind, dass auftretende Fehler sich nicht auf die anderen Einheiten auswirken dürfen.

Beziehungen - Grafik




Beziehungen - Detail

- **Untersuchung betroffener Subsysteme -> Units of Mitigation** : Gliederung der Einheiten der Fehlerbehandlung
  Die Units of Mitigation stellen Einheiten der Fehlerbehandlung dar. Wissen über Subsysteme, die als Bestandteile in die zu gestaltende verteilte Krankenakte eingehen, ist für die Definition dieser Einheiten und die daraus resultierende Gliederung des Gesamtsystems für die Fehlerbehandlung notwendig.
- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.
- **Error Containment Barrier -> Units of Mitigation** : Wirkungsbereich
  Eine Error Containment Barrier dient dazu, die Auswirkung eines Fehlers auf eine Unit of Mitigation zu beschränken.
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 92
- **Units of Mitigation -> Redundancy** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Units of Mitigation -> Quarantine** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Units of Mitigation -> Escalation** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden

Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.

- **Units of Mitigation -> Someone in Charge** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Units of Mitigation -> Software Update** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 35
- **Units of Mitigation -> Availability Requirements** : Einheiten für die Verfügbarkeitsanf. def. werden
  Durch die Anwendung des Units-of-Mitigation-Patterns wird das System in logische Einheiten für die Fehlerkompensation gegliedert. Jede dieser Einheiten trägt durch seine eigene Verfügbarkeit bzw. Fehlertoleranz zur Verfügbarkeit des Gesamtsystems bei.
- **Units of Mitigation -> Leaky Bucket Counter** : Überwachung ungewöhnlicher Fehlerhäufigkeiten
  Für jede Unit of Mitigation kann ein Leaky Bucket Counter angelegt werden. Der Leaky Bucket Counter dient dazu, bei ungewöhnlich häufigem Auftreten von Fehlern innerhalb kurzer Zeit einen Alarm auslösen zu können. Das kann z. B. durch versenden einer Nachricht etc. erfolgen.
  Quelle: Pattern for Fault Tolerant Software; Hanmer; S. 136

## *209. Unparochialness Requirements*

Quellen

**Software Requirement Patterns (Best Practices), Steven Withall**

S. 254-261

Stephen Withall, Software Requirement Patterns, Microsoft Press, 2007.
Online Beispiele: http://www.withallyourequire.com/reqtpatternsoverview.html

Schwerpunkt(e)

- Flexibilität

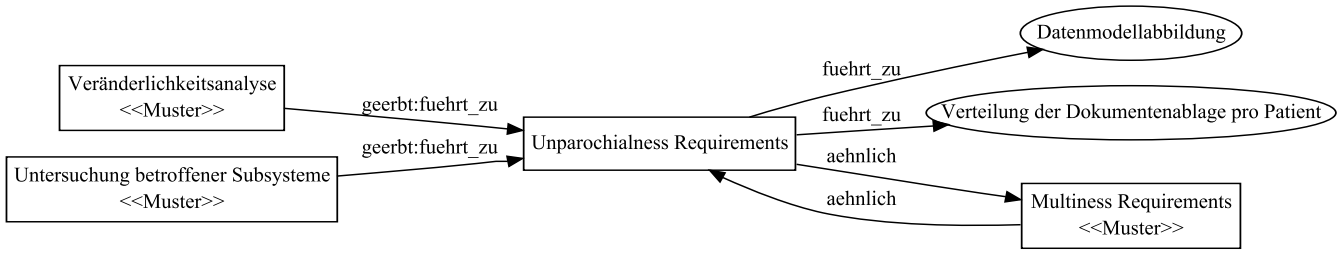
Gruppe(n)

- Ermittlung der Flexibilitätsanforderungen

Kurzbeschreibung

Das Unparochialness-Requirements-Pattern wird verwendet um die Anforderungen bezüglich der Fähigkeiten eines Systems zu spezifizieren, sich auf die Bedingungen verschiedener Umgebungen anzupassen. Es dient also der Beschreibung der Anpassbarkeit (also z.B. auch Konfigurierbarkeit) des Systems.

Beziehungen - Grafik




Beziehungen - Detail

- **Multiness Requirements -> Unparochialness Requirements** : entweder oder
  Quelle: Software Requirement Patterns; Withall; S. 261: *"Do not use the multiness requirement pattern when the system need support only one from a range of alternatives; use the unparochialness requirement pattern in that case"*
- **Unparochialness Requirements -> Datenmodell-Abbildung** : beeinflusst die Auswahl
  Das Unparochialness-Requirements-Pattern dient der Definition von Anforderungen bezüglich der Fähigkeit eines Systems, sich veränderlichen Umgebungsbedingungen anzupassen. Die verschiedenen Patterns der Gruppe Datenmodell-Abbildung führen zu unterschiedlich flexiblen Datenmodellen. Während z. B. das Entity-Attribute-Value-Pattern zu einem leicht veränderbaren Datenmodell führt, wird beim Domain-Model-Pattern das Datenmodell relativ statisch, dafür aber einfacher verwendbar, abgelegt.
- **Unparochialness Requirements -> Verteilung der Dokumentenablage pro Patient** : schränken ein
  Gewählte Unparochialness Requirements schränken die Menge der möglichen Verteilungslösungen ein. Abgeleitet aus (Withall, S. 260 Punkt 5, External Systems) -> Anforderungen an Kommunikationsfähigkeiten zu externen Systemen.
- **Unparochialness Requirements -> Multiness Requirements** : entweder oder
  Quelle: Software Requirement Patterns; Withall; S. 261: *"Do not use the multiness requirement pattern when the system need support only one from a range of alternatives; use the unparochialness requirement pattern in that case"*
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Geerbte Beziehung: fuehrt_zu_beziehung
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.

## *210. Untersuchung betroffener Subsysteme*

Englische Bezeichnung

- Analysis of Relevant Subsystems

Schwerpunkt(e)

- Kommunikation
- Zuverlässigkeit
- Flexibilität

Gruppe(n)

- Analyse flexibilitätsbeeinflussender Faktoren
- Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl.
- Analyse des Kommunikationsbedarfs

Kontext
Es wurde bereits eine Analyse der organisatorischen Verteilung im Verbund sowie eine Analyse der betroffenen Behandlungspfade durchgeführt. Bei der Analyse der organisatorischen Verteilung wurden mindestens mehrere eigenständige Versorgungseinrichtungen oder eine Einrichtung mit einer Vielzahl von Organisationseinheiten mit unterschiedlichem Leistungsspektrum ermittelt. Die Analyse der betroffenen Behandlungspfade zeigt, dass diese Pfade Einrichtungs- bzw. Organisationseinheitsübergreifend strukturiert sind.

Problem
Es ist wenig wahrscheinlich, im Gesundheitswesen auf eine Landschaft ohne bereits existierende und im Einsatz befindliche IT-Systeme zu stoßen. Deshalb ist es notwendig Informationen über diese bereits existierenden IT-Systeme (fortfolgend Subsysteme genannt) zu Sammeln um ein geeignetes Architekturkonzept für eine verteilte Krankenakte entwickeln zu können.

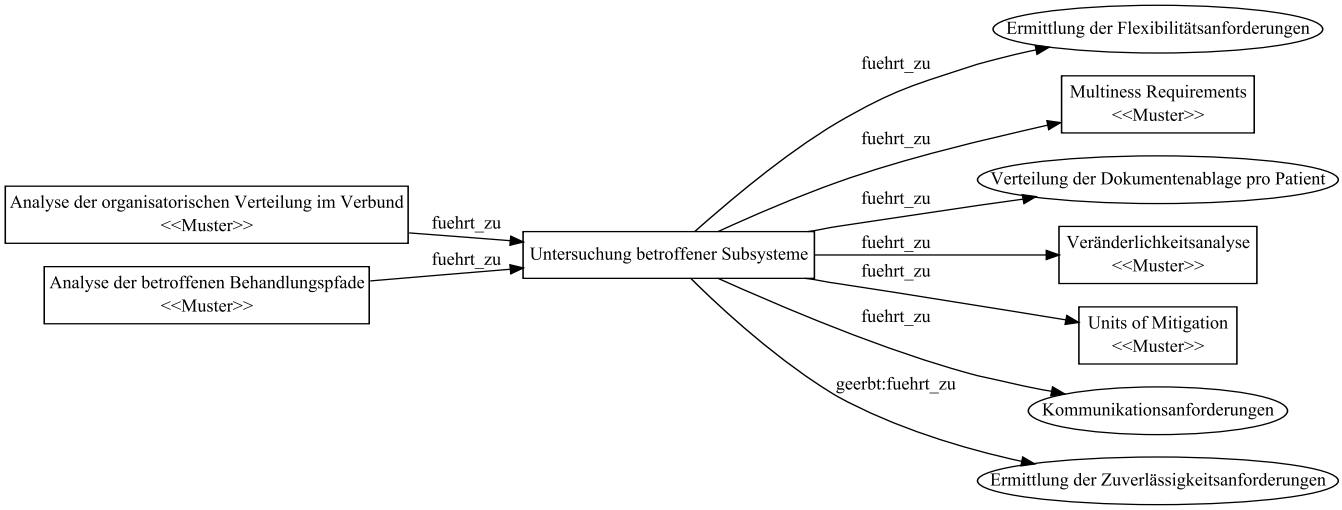
Lösung

Schritt 1:
Ausgehend von der Dokumentation zu den ermittelten Behandlungspfaden werden die an den darin beschriebenen Aktionen beteiligten IT-Systeme ermittelt und in Form einer Liste erfasst.

Schritt 2:
Jedes der erfassten Subsysteme wird einzeln untersucht und beschrieben. Dabei werden einerseits, die enthaltenen und für die verteilte Krankenakte relevanten Daten durch eine Auflistung geeigneter Klassennamen festgehalten. Außerdem werden Schnittstellen, über die andere Systeme mit dem untersuchten System kommunizieren können, erfasst. Implementiert ein Subsystem ein oder mehrere medizinische IT-Standards, so wird das ebenfalls festgehalten.

Ergebnis:
Das Ergebnis der Anwendung dieses Patterns ist eine Auflistung der am Zweck der verteilten Krankenakte beteiligten bereits existierenden IT-Systeme einschließlich einer Auflistung von Klassen enthaltener Daten bzw. Standards sowie der Schnittstellen über welche die enthaltenen Daten durch andere Systeme erfragt bzw. geändert werden können.

Beziehungen - Grafik




Beziehungen - Detail

- **Analyse der organisatorischen Verteilung im Verbund -> Untersuchung betroffener Subsysteme** : Input: Einrichtungen mit Subsystemen
  Im Rahmen der Analyse der organisatorischen Verteilung werden an der verteilten Krankenakte beteiligte Einrichtungen ermittelt. Die IT-Systeme dieser Einrichtungen werden im Rahmen der Untersuchung betroffener Subsysteme untersucht.
- **Analyse der betroffenen Behandlungspfade -> Untersuchung betroffener Subsysteme** : Input: KA-relevante Aufgaben
  Die Analyse der betroffenen Behandlungspfade liefert für die Untersuchung der betroffenen Subsysteme die Menge der für die Entwicklung der verteilten Krankenakte relevanten Aufgaben. Relevant sind dabei alle Aufgaben, die in den Behandlungspfaden, die durch die verteilte Krankenakte unterstützt werden sollen, enthalten sind.
- **Untersuchung betroffener Subsysteme -> Ermittlung der Flexibilitätsanforderungen** : Input: Schnittstellen, Synchronisationsbedarf usw.
  Bei der Untersuchung betroffener Subsysteme werden Systeme identifiziert, die als Subsysteme teil der verteilten Krankenakte werden sollen. Informationen über diese Systeme bilden eine zentrale Basis für die Ermittlung der Flexibilitätsanforderungen.
- **Untersuchung betroffener Subsysteme -> Multiness Requirements** : beeinflusst direkt
  Das Ergebnis der Untersuchung betroffener Subsysteme beeinflusst direkt die Anforderungen an die Vielfältigkeit der Schnittstellen.
- **Untersuchung betroffener Subsysteme -> Verteilung der Dokumentenablage pro Patient** : Verwendbarkeit der Subsysteme

- **Untersuchung betroffener Subsysteme -> Veränderlichkeitsanalyse** : Input: Liste der Subsysteme
  Die Untersuchung der betroffenen Subsysteme liefert für die Veränderlichkeitsanalyse als Ausgangsdatenbasis eine Auflistung der verschiedenen von der Entwicklung einer verteilten Krankenakte betroffenen Subsysteme einschließlich ihrer relevanten Datenbestände, Schnittstellen und implementierten Standards.
- **Untersuchung betroffener Subsysteme -> Units of Mitigation** : Gliederung der Einheiten der Fehlerbehandlung
  Die Units of Mitigation stellen Einheiten der Fehlerbehandlung dar. Wissen über Subsysteme, die als Bestandteile in die zu gestaltende verteilte Krankenakte eingehen, ist für die Definition dieser Einheiten und die daraus resultierende Gliederung des Gesamtsystems für die Fehlerbehandlung notwendig.
- **Untersuchung betroffener Subsysteme -> Kommunikationsanforderungen** : Input: Datenquellen, Schnittstellen und Standards
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.

## *211. Validation Requirements*

Schwerpunkt(e)

- Zuverlässigkeit

Gruppe(n)

- Ermittlung der Zuverlässigkeitsanforderungen

Kontext

Bei der Eingabe von Daten in ein System können durch den eingebenden Akteur Fehler gemacht werden. Auch bei der Übertragung von Daten aus anderen Systemen können fehlerhafte Daten übermittelt werden.

Problem
Welche dieser Fehler müssen systemseitig vermieden werden, ohne dass eine zusätzliche Kontrolle durch dritte Benutzer (vgl. Approval Requirements) erforderlich ist.

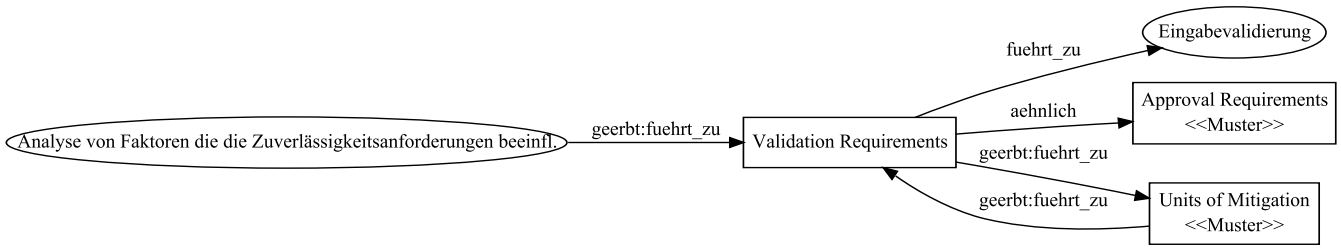
Lösung

Validierungsanforderungen (Validation Requirements) können und sollten systematisch erfasst werden. Eine Validierungsanforderung muss mindestens die folgenden Punkte enthalten:

- Das Datenelement, das validiert werden soll,
- relevante Stellen, an denen dieses Datenelement in der Oberfläche bzw. an den Schnittstellen des Systems vorkommt und
- die Regel anhand der die Gültigkeit des Elements geprüft werden kann

Regeln zur Validierung basieren meist auf fachlichen Zusammenhängen zwischen den beteiligten Objekten und können als formale Bedingungen (z. B. mittels Object Constraint Language (OCL)) beschrieben werden. Die Verwendung von OCL setzt die Existenz eines Klassen-/Entitäten- oder Objektmodells voraus.

Beziehungen - Grafik




Beziehungen - Detail

- **Validation Requirements -> Eingabevalidierung** : Bereich: Dateneingabe
  Ein Bereich der Validation Requirements befasst sich mit der Validierung von Daten während deren Eingabe bzw. Übertragung an das System, in dem die Validierung durchgeführt wird.
- **Validation Requirements -> Approval Requirements** :
  Approval Requirements beschreiben die Anforderungen an eine Software bezüglich der Prüfung eingegebener Daten durch Benutzer, z. B. im Rahmen eines Approval Workflow, während Validation Requirements die Anforderungen an eine automatische Überprüfung (Validierung) eingegebener Daten zum Zeitpunkt der Eingabe mittels Regeln beschreiben.
- **Analyse von Faktoren die die Zuverlässigkeitsanforderungen beeinfl. -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einflussfaktoren
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Menge der möglichen, also an die zukünftige Applikation stellbaren Anforderungen bezüglich der Zuverlässigkeit variiert abhängig von verschiedenen Faktoren, die gleichsam als Rahmenbedingungen für die Definition der Anforderungen wirken. So beeinflusst beispielsweise eine organisatorisch bereits bestehende Verteilungssituation direkt die Menge der zu erwartenden Systembestandteile und somit auch die für diese Bestandteile zu definierenden Anforderungen.

- **Units of Mitigation -> Ermittlung der Zuverlässigkeitsanforderungen** : Input: Einheiten für die Zuverlässigkeitsanforderungen definiert werden
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Ermittlung der Zuverlässigkeitsanforderungen wird zweimal durchgeführt. Einmal mit dem Gesamtsystem und ein weiteres Mal für jede der, für das Gesamtsystem ermittelten Units of Mitigation. Die hier vorliegende Beziehung bildet den Fall der zweiten Ermittlung der Zuverlässigkeitsanforderungen für die einzelnen Units of Mitigation ab.
- **Ermittlung der Zuverlässigkeitsanforderungen -> Units of Mitigation** : Input: Zuverlässigkeitsanforderungen für das Gesamtsystem
  Geerbte Beziehung: fuehrt_zu_beziehung
  Auf Basis der Zuverlässigkeitsanforderungen für das Gesamtsystem werden die Units of Mitigation, also die im Fehlerfall zusammenhängend reagierenden Untereinheiten der Fehlerbehandlung definiert.

## *212. Veränderlichkeitsanalyse*

Englische Bezeichnung

- Analysis of the Propability of Change

Schwerpunkt(e)

- Flexibilität

Gruppe(n)

- Analyse flexibilitätsbeeinflussender Faktoren

Kontext

Die Ergebnisse der Analyse der betroffenen Behandlungspfade und der Untersuchung betroffener Subsysteme liegen vor.

Problem
Die Umgebung einer verteilten Krankenakte ist nicht statisch, durch Veränderungen in den Verbünden der Versorgungseinrichtungen, technische Neuerungen oder auch die turnusmäßige Aktualisierung vorhandener IT-Systeme können Veränderungen an der Software der verteilten Krankenakte notwendig sein. Da Software in der Regel besser mit erwarteten Veränderungen zurecht kommt als mit unerwarteten ist es vorteilhaft, diese Veränderungen bereits bei der Spezifikation der Anforderungen und dadurch bei der Konzeption des Systems zu berücksichtigen.

Lösung
Um zu gewährleisten, dass sich die Architektur der verteilten Krankenakte dazu eignet, erwartbare Veränderungen durch vorbestimmte Mechanismen zu ermöglichen, ist es

notwendig, dass die Arten der erwartbaren Veränderungen bereits Bestandteil der Anforderungen an ein zu entwickelndes System sind.

### Typische Veränderungen

Veränderungen die für alle verteilten Krankenakten mit einrichtungsübergreifenden Behandlungspfaden gelten sind:

- die generelle Veränderlichkeit des Ablaufs eines Behandlungspfads innerhalb der in Kapitel 2.3.2. beschriebenen Grenzen,
- die Einführung neuer oder anderer Schnittstellenstandards,
- der Ein- bzw. Ausstieg von Einrichtungen in bzw. aus einem Verbund,
- der Austausch eines Subsystems durch eine Einrichtung des Verbunds und
- die Veränderung technischer Eigenschaften von Subsystemen z. B. durch Versionswechsel.

### Durchführung

Die Veränderlichkeitsanalyse wird für jedes identifizierte Subsystem und jeden Behandlungspfad einzeln durchgeführt. Bei der Analyse soll festgestellt werden, welche Veränderungen bei welchem Subsystem bzw. Pfad zu erwarten sind und welche ausgeschlossen werden können.

### Ergebnis

Ergebnis der Veränderlichkeitsanalyse ist sowohl eine Auflistung von Veränderlichkeits-potenzialen nach Subsystemen und Pfaden als auch eine aggregierte Darstellung, aus der die verschiedenen Flexibilitätsanforderungen für das Softwaresystem der verteilten Krankenakte entwickelt werden können.

### Vorbemerkung zum Beispiel:

Abhängig von der Systemarchitektur des untersuchten Subsystems sind Änderungen in diesen Systemen unterschiedlich wahrscheinlich. Handelt es sich bei dem Subsystem um eine Eigenentwicklung der betreibenden Versorgungseinrichtung, so kann diese über Veränderungen selbst bestimmen. Handelt es sich um ein Produkt, das durchgängig einen bestimmten Standard implementiert, wie z. B. ein auf dem DICOM-Standard basierendes PACS, so ist ein Wechsel des verfügbaren Kommunikationsstandards weniger wahrscheinlich, als das bei einem System der Fall ist, das zwar Schnittstellen eines oder mehrerer Standards anbietet, aber intern unabhängig davon entwickelt ist.

### Beispiel: PACS

Eines der untersuchten Subsysteme ist ein PACS, das in allen seinen Ausprägungen entsprechend des DICOM-Standards implementiert ist. Das PACS wird bei der beteiligten Versorgungseinrichtung durch ein eigens entwickeltes System zur Abbildung des Röntgen-Workflows bedient und von der untersuchenden Modalität direkt per C-MOVE mit den erstellten Bildern befüllt. Welche Veränderungen sind zu erwarten:

- Häufig:
    - Modalitäten, die ihre Bilder in einer abweichenden Transfersyntax (mit z. B. anderem Bildformat innerhalb der DICOM-Datei) im PACS speichern.

- Gelegentlich:
  - Änderungen am System für den Röntgen-Workflow
  - Änderung von Parametern für die Kommunikation mit dem PACS (z. B. IP-Adresse oder DNS-Name des DICOM-Servers)
- Selten:
  - Austausch des PACS durch ein Archiv eines anderen Herstellers

Beziehungen - Grafik


Beziehungen - Detail

- **Analyse der betroffenen Behandlungspfade -> Veränderlichkeitsanalyse** : Input: Behandlungspfade
  Behandlungspfade können, vor allem wenn sie einrichtungsübergreifend definiert sind, häufigen Änderungen unterworfen sein. Typische Änderungsgründe für einrichtungsübergreifende Behandlungspfade sind die Veränderung der vertraglichen Verbindungen zwischen den Einrichtungen oder die Einführung neuer Behandlungsmethoden. Das Pattern "Analyse veränderlicher Bestandteile" dient dazu zu ermitteln, welche Bestandteile der Anforderungen häufigen Veränderungen unterworfen sind und welche als relativ konstant betrachtet werden können.
- **Untersuchung betroffener Subsysteme -> Veränderlichkeitsanalyse** : Input: Liste der Subsysteme
  Die Untersuchung der betroffenen Subsysteme liefert für die Veränderlichkeitsanalyse als Ausgangsdatenbasis eine Auflistung der verschiedenen von der Entwicklung einer verteilten Krankenakte betroffenen Subsysteme einschließlich ihrer relevanten Datenbestände, Schnittstellen und implementierten Standards.
- **Veränderlichkeitsanalyse -> Ermittlung der Flexibilitätsanforderungen** : Input: Wahrscheinliche Veränderungen
  Die Veränderlichkeitsanalyse liefert für die Ermittlung der verschiedenen Flexibilitätsanforderungen eine Auflistung von verschiedenen wahrscheinlich zu erwartenden Veränderungen sowohl allgemein für das Gesamtsystem als auch bezogen auf die verschiedenen Subsysteme bzw. Behandlungspfade.

## *213. Visitor*

Quellen

**Design Patterns, Gamma, Helm, Johnson, Vlissides**

S. 331-344

Design Patterns; Elements of Reusable Object-Oriented Software; Gamma, Helm, Johnson, Vlissides; Addison-Wesely, 38. Auflage, Westford, Massachusetts, July 2010

Schwerpunkt(e)

- Flexibilität

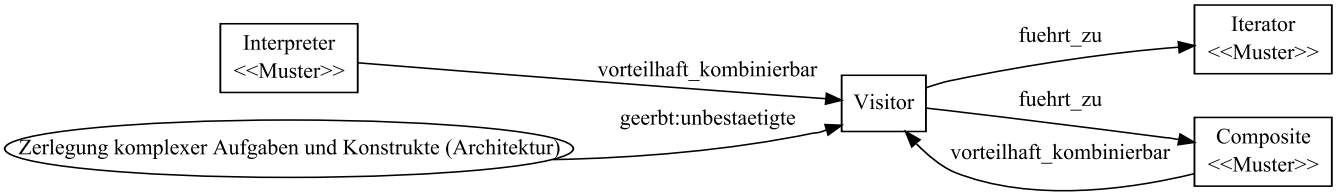
Gruppe(n)

- Zerlegung komplexer Aufgaben und Konstrukte (Design)
- Sonstige flexibilitätserhöhende Design-Patterns

Kurzbeschreibung

Das Pattern "Visitor" beschreibt einen Ansatz, Methoden außerhalb der Klasse zu definieren, auf deren Objekte sie angewendet werden. Dieses Vorgehen kann z. B. genutzt werden, um allgemeinere Objektstrukturen zu erweitern, ohne deren ursprüngliche Übersichtlichkeit zu verschlechtern.

Beziehungen - Grafik




Beziehungen - Detail

- **Composite -> Visitor** : Zusaetzliche Operationen hinzufuegen
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Interpreter -> Visitor** : adding operation
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
  Quelle: Design Patterns; Gamma; S. 255; Related Patterns;
  *"Visitor can be used to maintain the behavior in each node in the abstract syntax tree in one class."*
- **Visitor -> Iterator** : defining traversals
  Quelle: Design Patterns; Gamma et al.; letzte Seite; Design Pattern Relationships
- **Visitor -> Composite** : Bestandteil der Umsetzung
  Quelle: Design Patterns; Gamma et al; S. 334-335
  Das Visitor Pattern hat die Bestandteile Visitor; ConcreteVisitor; Element; ConcreteElement und ObjectStructure. Der Bestandteil ObjectStructure hat folgende Eigenschaft: "*ObjectStructure* [...] *may either be a composite or a collection such as a list or a set."*
- **Zerlegung komplexer Aufgaben und Konstrukte (Architektur) -> Zerlegung komplexer Aufgaben und Konstrukte (Design)** : Konkretisierung der Umsetzung
  Geerbte Beziehung: unbestaetigte_beziehung
  Die Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Architektur)" können unter Zuhilfenahme anderer Muster umgesetzt werden. Muster der Gruppe "Zerlegung komplexer Aufgaben und Konstrukte (Design)" dienen ebenso

zur Zerlegung komplexer Sachverhalte, aber auf deutlich feingranularerer Ebene. Manche der Patterns der Architektur-Ebene lassen sich allerdings direkt durch Anwendung entsprechender Design-Patterns umsetzen. (Beispiel: Pipes and Filters -> Chain of Responsibility)

## *214. Voting*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 118-121

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

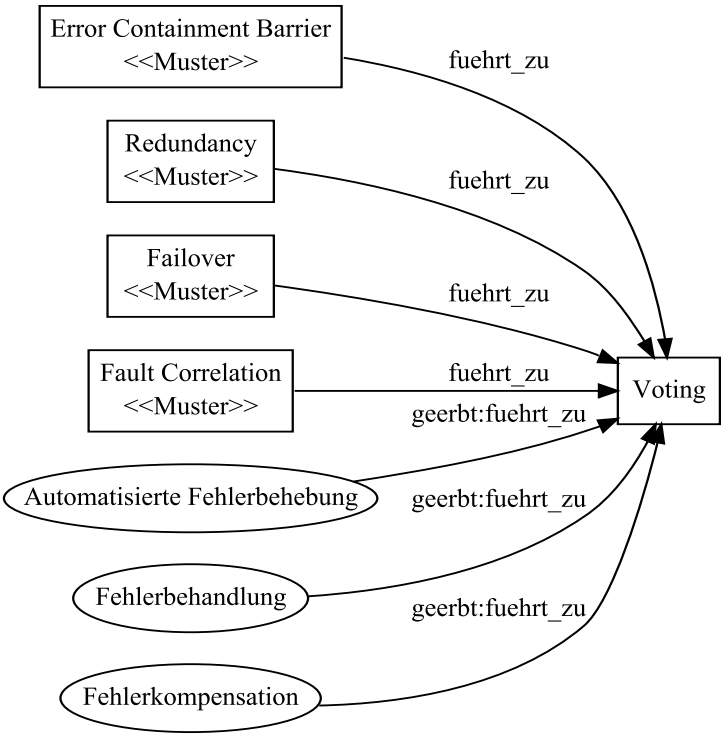
Gruppe(n)

- Fehlererkennung

Kurzbeschreibung

Das Pattern "Voting" befasst sich mit der Frage, welche Lösung als richtig erachtet werden soll, wenn mehrere redundante Implementierungen unterschiedliche Ergebnisse liefern.

Beziehungen - Grafik



Beziehungen - Detail

- **Error Containment Barrier -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Redundancy -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Failover -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Fault Correlation -> Voting** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.
- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *215. Vulnerability Assessment*

Quellen

**Security Patterns, Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad**

S. 125-136

Security Patterns; Integrationg Security and Systems Engineering; Schumacher, Fernandez-Buglioni, Hybertson, Buschmann, Sommerlad; Wiley; 2006

Schwerpunkt(e)

- Sicherheit

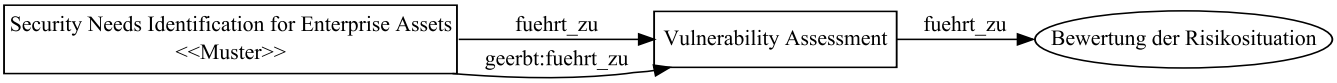
Gruppe(n)

- Analyse der Sicherheitssituation

Kurzbeschreibung

Das Pattern "Vulnerability Assessment" beschreibt ein Verfahren zur Verwundbarkeitsanalyse.

Beziehungen - Grafik




Beziehungen - Detail

- **Security Needs Identification for Enterprise Assets -> Vulnerability Assessment** : Identifikation von Schwächen
  Quelle: Security Patterns; Schumacher et al.; S. 60
- **Vulnerability Assessment -> Bewertung der Risikosituation** : Input: Schwachstellen
  Quelle: S. 60 und 61 in Security Patterns, Schumacher et al.: Die in der Quelle abgebildete Beziehung baut ursprünglich direkt auf Risk determination auf. Durch die Einführung der Gruppe Bewertung der Risikosituation wird die Beziehung zur Gruppe hin übernommen.
  Quelle: S. 147 in Security Patterns, Schumacher et al.;Risk Determination, Liabilities: "*The results are based on the completeness and subjectivity of Asset Valuation, Threat Assessment and Vulnerability Assessment* [...]."
- **Security Needs Identification for Enterprise Assets -> Analyse der Sicherheitssituation** : Grobanalyse zu Detailanalyse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Security-Needs-Identification-for-Enterprise-Assets-Pattern dient der Identifikation und Bewertung gefährdeter Güter, Anlagen, Personen und Daten in den untersuchten medizinischen Versorgungseinrichtungen. Die ermittelten Gefährdeten, sowie das ebenfalls ermittelte Einflusspotenzial auf den Gesamtbetrieb dienen dazu im Rahmen der Analyse der Sicherheitssituation ermittelte Sicherheitsrisiken und andere Einflussfaktoren nach ihrer Wichtigkeit zu bewerten.

## *216. Watchdog*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 107-109

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

### Schwerpunkt(e)

- Zuverlässigkeit

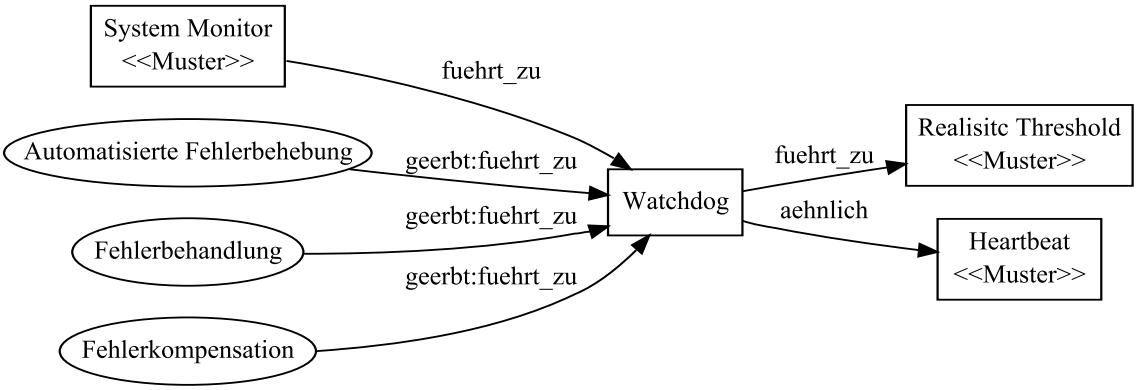
### Gruppe(n)

- Fehlererkennung

### Kurzbeschreibung

Das Pattern "Watchdog" (Wachhund oder besser Wächter) beschreibt ein einfaches und leichtgewichtiges Verfahren zur Überwachung der Verfügbarkeit.

### Beziehungen - Grafik




### Beziehungen - Detail

- **System Monitor -> Watchdog** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Watchdog -> Realisitc Threshold** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; S. 87
- **Watchdog -> Heartbeat** : Ähnliche Zielsetzung
  Quelle: Patterns for Fault tolerant Software; Hanmer; S. 107; Bei Heartbeat senden Komponenten Nachrichten über ihren Zustand an einen zentralen System Monitor. Bei Watchdog werden die Komponenten von einer Überwachungskomponente aktiv überwacht. Die Watchdog Komponente platziert sich selbst zwischen den zu überwachenden Komponenten und dem System Monitor.
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.
- **Fehlerbehandlung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen

Behandlung. Um Muster zur Behandlung von Fehlern verwenden zu können, ist es notwendig auch passende Muster zur Fehlererkennung auszuwählen und anzuwenden.

- **Fehlerkompensation -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung
  Zu erkennen, dass ein Fehler vorliegt, ist eine notwendige Voraussetzung, um gezielte Maßnahmen zur Milderung oder Kompensation von dessen Auswirkungen einzuleiten.

## *217. What to save*

Quellen

**Patterns for Fault Tolerant Software, Hanmer**

S. 169-170

Patterns for Fault Tolerant Software, Robert S. Hanmer, Wiley Series in Software Design Patterns, 2007

Schwerpunkt(e)

- Zuverlässigkeit

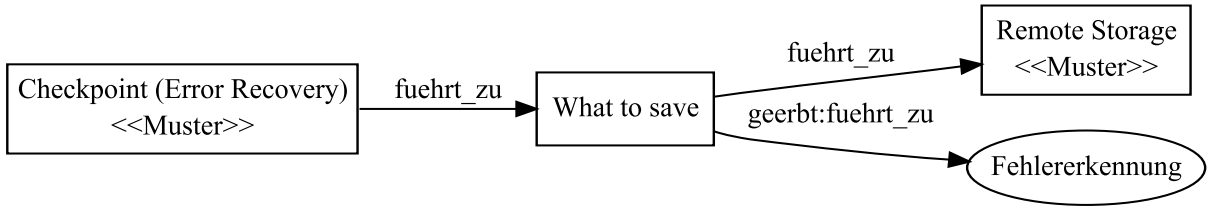
Gruppe(n)

- Automatisierte Fehlerbehebung

Kurzbeschreibung

Das Pattern "What to save" beschäftigt sich mit der Frage, was in einem Checkpoint (Error Recovery) gesichert werden soll.

Beziehungen - Grafik




Beziehungen - Detail

- **Checkpoint (Error recovery) -> What to save** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **What to save -> Remote Storage** :
  Quelle: Patterns for Fault Tolerant Software; Hanmer; letzte Seite; A Pattern Language for Fault Tolerant Software
- **Automatisierte Fehlerbehebung -> Fehlererkennung** : notwendige Voraussetzung
  Geerbte Beziehung: fuehrt_zu_beziehung

Die Erkennung eines Fehlers ist die notwendige Voraussetzung für dessen automatische Behebung. Werden Patterns zur Fehlerbehebung verwendet, so ist dazu auch die Verwendung von zum Anwendungsfall passenden Patterns zur Fehlererkennung notwendig.

## *218. Zentrale Autorisierungsregeln*

Englische Bezeichnung

- Centralized Authorization Rules

Schwerpunkt(e)

- Sicherheit

Gruppe(n)

- Verteilung der Autorisierungsinformation

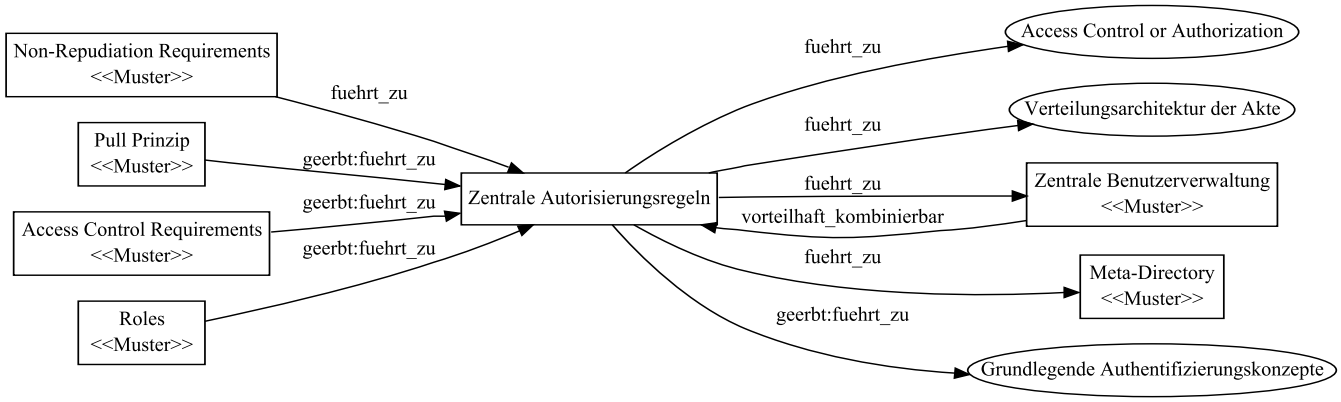
Kontext

In einem System mit verteilter Ablage von Patientendaten müssen Autorisierungsregeln verwaltet werden.

Problem
Die Anzahl der beteiligten Subsysteme ist groß. Die Verwendung von ausschließlich lokalen Autorisierungsregeln führt zu einem komplexen, durch seine verteilte Natur schwer überschaubaren Regelwerk. Aufgrund der Komplexität und fehlenden Übersichtlichkeit liegt eine erhöhte Fehlerwahrscheinlichkeit vor.

Lösung
Es gibt eine zentrale Stelle, an der die Autorisierungsregeln für alle beteiligten datenhaltenden Systemknoten bereitstehen. Durch diese zentrale Verwaltung von Autorisierungsregeln ist es möglich die Berechtigungen eines Benutzers zu einem Zeitpunkt exakt zu bestimmen und in einem zentralen Werkzeug darzustellen. Außerdem ermöglicht dieser Ansatz auch eine Historisierung von den Berechtigungen der Benutzer.

Beziehungen - Grafik




Beziehungen - Detail

- **Non-Repudiation Requirements -> Zentrale Autorisierungsregeln** :
  Dokumentation früherer Berechtigungen
  Durch zentral verwaltete Autorisierungsregeln wird die Historisierung der Berechtigungen von Benutzer möglich bzw. erleichtert.
- **Zentrale Benutzerverwaltung -> Zentrale Autorisierungsregeln** :
  Es ist vorteilhaft, eine zentrale Benutzerverwaltung mit zentral verwalteten Autorisierungsregeln zu kombinieren, da so eine zentrale Komponente zu jedem Zeitpunkt des Systembetriebs Auskunft über die exakten Berechtigungen eines Benutzers erteilen kann. Das ermöglicht sowohl eine Benutzer- und Rechtehistorisierung als auch einen verbesserten Überblick bei der Verwaltung von Rechten.
- **Zentrale Autorisierungsregeln -> Access Control or Authorization** : zur Umsetzung ohne verteilte Regeln
  Aktenbezogene Autorisierungskonzepte lassen sich durch Kombination von Patterns der Gruppe Access Control or Authorization umsetzen.
  Bei der Verwendung zentraler Autorisierungsregeln ist eine Berücksichtigung des Verteilungsaspekts nicht notwendig, da die Autorisierungsinformation nicht verteilt vorliegt.
- **Zentrale Autorisierungsregeln -> Verteilungsarchitektur der Akte** : kombinierbar mit beliebiger Verteilungsarchitektur
  Prinzipiell sind zentrale Autorisierungsregeln beliebig mit Verteilungsarchitekturen der Akte kombinierbar, sofern alle beteiligten Knoten einer Architektur mit nicht zentraler Datenspeicherung geeignet sind einen gemeinsamen Dienst zur Verwaltung der Autorisierungsregeln zu verwenden. Da das bei Umgebungen mit einer Vielzahl bereits bestehender Anwendungen selten der Fall ist, existiert die Kombinierbarkeit vorrangig mit den Ausprägungen "Akte mit regional zentralisierter Datenspeicherung" und "Akte mit zentraler Datenspeicherung".
- **Zentrale Autorisierungsregeln -> Zentrale Benutzerverwaltung** : Benötigt
  Zur Umsetzung zentral verwalteter Autorisierungsregeln sind über alle Systemknoten hinweg eindeutige Benutzer notwendig um das zugrundeliegende Regelwerk eindeutig

zu halten. Aus diesem Grund wird zur Umsetzung zentraler Autorisierungsregeln entweder das zentrale Benutzerverwaltung- oder das Meta-Directory-Pattern benötigt.

- **Zentrale Autorisierungsregeln -> Meta-Directory** : Benötigt
  Zur Umsetzung zentral verwalteter Autorisierungsregeln sind über alle Systemknoten hinweg eindeutige Benutzer notwendig um das zugrunde liegende Regelwerk eindeutig zu halten. Aus diesem Grund wird zur Umsetzung zentraler Autorisierungsregeln entweder das zentrale Benutzerverwaltung- oder das Meta-Directory-Pattern benötigt.
- **Pull Prinzip -> Verteilung der Autorisierungsinformation** : Gewährleistung des Zugriffsschutzes
  Geerbte Beziehung: fuehrt_zu_beziehung
  Erfolgt der Transport der Akteninhalte nach dem Pull-Prinzip, so ist die verteilte oder zentrale Verwaltung von Autorisierungsregeln zwangsläufig notwendig um beim Zugriff den Schutz vor unberechtigtem Zugriff zu gewährleisten.
- **Access Control Requirements -> Aktenbezogene Autorisierungskonzepte** : Entscheidung für oder gegen die Verteilung des Autorisierungsmechanismus
  Geerbte Beziehung: fuehrt_zu_beziehung
  Die spezifizierten Access Control Requirements beeinflussen die Verteilbarkeit von Autorisierungsregeln und dienen als so Informationsbasis für die Auswahl des geeigneten Patterns aus der Gruppe Verteilung der Autorisierungsinformation.
- **Roles -> Verteilung der Autorisierungsinformation** :
  Geerbte Beziehung: fuehrt_zu_beziehung
  Das Roles-Pattern beschreibt die Ermittlung und Definition von Rollen und zugehörigen Berechtigungen für Benutzer eines Systems. Im komplexen Umfeld einer verteilten Krankenakte haben Benutzer diese Rollen aber nur bezogen auf bestimmte Organisationen oder Organisationseinheiten. Bei der Umsetzung aktenbezogener Autorisierungskonzepte mit zentral verwalteten Autorisierungsregeln ist deshalb auch die Berücksichtigung des Organisationsbezugs von Rollen und Rechten zu bedenken.
- **Verteilung der Autorisierungsinformation -> Grundlegende Authentifizierungskonzepte** : Benutzeridentifikation
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um die Gültigkeit von Autorisierungsbedingungen bei einem Funktionsaufruf prüfen zu können, ist die Identifikation des zu Berechtigenden mittels Authentifizierung notwendig.

## *219. Zentrale Benutzerverwaltung*

Englische Bezeichnung

- Centralized User Management

Schwerpunkt(e)

- Sicherheit

### Gruppe(n)

- Anzahl der Authentifizierungsquellen

### Kontext

In einem System aus n beteiligten Knoten soll gemeinsam gearbeitet werden. Für die Durchführung der Arbeit ist eine Autorisierung notwendig. Aus diesem Grund müssen die beteiligten Benutzer authentifiziert werden.

### Problem

Wenn jeder Knoten seine Benutzer selbst verwaltet sind bei der Umsetzung von Behandlungsprozessen die den Einflussbereich mehrerer Knoten betreffen redundante Benutzerkonten notwendig. Für jeden am Prozess beteiligten Benutzer müssen in allen beteiligten Knoten Konten angelegt werden.

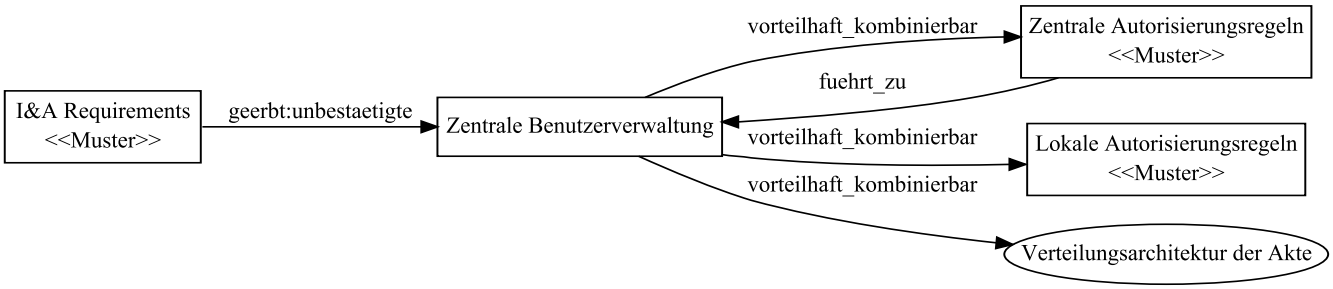
### Lösung

Die Lösung für dieses Problem ist die Einführung einer zentralen Komponente, die die Menge der Benutzer aller Subsysteme beinhaltet und von den Subsystemen zur Durchführung der Authentifizierung verwendet wird.

### Beispiel

Typische Beispiele sind Datenbanken oder LDAP-Verzeichnisse die von vielen Systemen gemeinsam zur Benutzerauthentifizierung verwendet werden.

### Beziehungen - Grafik




### Beziehungen - Detail

- **Zentrale Autorisierungsregeln -> Zentrale Benutzerverwaltung** : Benötigt
  Zur Umsetzung zentral verwalteter Autorisierungsregeln sind über alle Systemknoten hinweg eindeutige Benutzer notwendig um das zugrundeliegende Regelwerk eindeutig zu halten. Aus diesem Grund wird zur Umsetzung zentraler Autorisierungsregeln entweder das zentrale Benutzerverwaltung- oder das Meta-Directory-Pattern benötigt.
- **Zentrale Benutzerverwaltung -> Zentrale Autorisierungsregeln** :
  Es ist vorteilhaft, eine zentrale Benutzerverwaltung mit zentral verwalteten Autorisierungsregeln zu kombinieren, da so eine zentrale Komponente zu jedem Zeitpunkt des Systembetriebs Auskunft über die exakten Berechtigungen eines Benutzers erteilen kann. Das ermöglicht sowohl eine Benutzer- und

Rechtehistorisierung als auch einen verbesserten Überblick bei der Verwaltung von Rechten.

- **Zentrale Benutzerverwaltung -> Lokale Autorisierungsregeln** :
  Eine zentrale Benutzerverwaltung kann auch mit lokalen Autorisierungsregeln, also Autorisierungsregeln die z. B. pro Knoten einer verteilten Krankenakte verwaltet werden, kombiniert werden. Vorteil bei dieser Kombination ist, dass für alle im Gesamtsystem verteilt existierenden Berechtigungen Regeln auf Basis von eindeutigen Benutzern existieren.
- **Zentrale Benutzerverwaltung -> Verteilungsarchitektur der Akte** : Abhängig von Wie sinnvoll eine zentrale Benutzerverwaltung ist und mit welchem Aufwand sie sich realisieren lässt, ist abhängig von der gewählten Verteilungsarchitektur der Akte.
- **I&A Requirements -> Anzahl der Authentifizierungsquellen** : Beeinflusst die Auswahl
  Geerbte Beziehung: unbestaetigte_beziehung

## *220. Zentraler Suchdienst*

Englische Bezeichnung

- Central Service for Searching

Schwerpunkt(e)

- Zuverlässigkeit
- Flexibilität

Gruppe(n)

- Suche im verteilten Dokumentensystem

Kontext

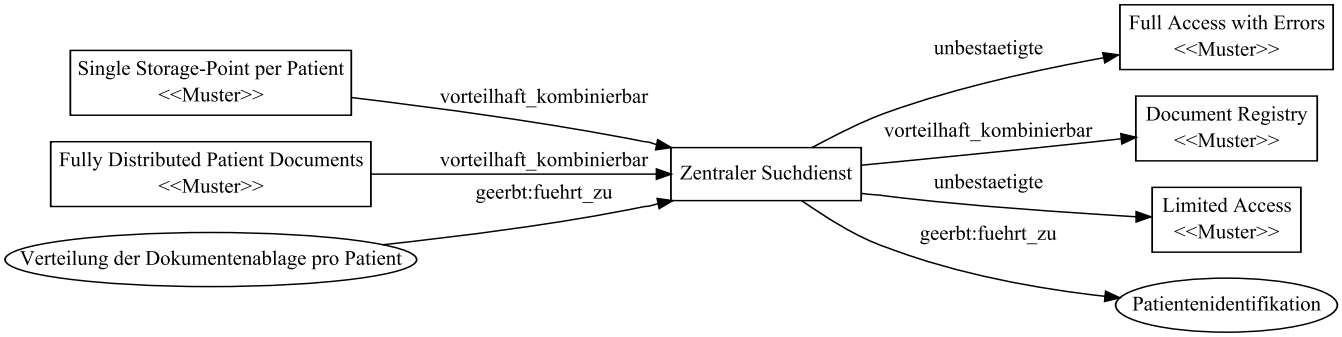
Im Rahmen der medizinischen Versorgung eines Patienten ist es an verschiedenen Stellen des Versorgungsprozesses nötig, Informationen über die medizinische Vorgeschichte eines Patienten einzuholen. Dazu ist eine effiziente Funktionalität zur Ermittlung relevanter Inhalte innerhalb eines komplexen Dokumentationssystems notwendig.

Problem
Dokumente oder Patienten können verteilt auf viele Systemknoten abgelegt sein. Eine Entität zu finden ist mit Aufwand verbunden. Aus diesem Grund ist für den Betrieb einer verteilten Krankenakte ein Mechanismus zur effizienten Auffindung solcher Inhalte notwendig.

Lösung
Eine mögliche Lösung des Problems der Findung von Inhalten in einer verteilten Krankenakte ist der Aufbau eines zentralen Suchdiensts. Dabei handelt es sich um eine Komponente des Systems, die Suchvorgänge über alle Knoten des Systems hinweg zentral koordiniert. Um die Belastung für die einzelnen Systemknoten geringer zu halten, als das bei Flooding basierter

Suche der Fall ist, wird in der Komponente durch die Verwendung des Registry-Patterns ein Index aufgebaut, der pro Suchvorgang den gezielten Ausschluss von sicher nicht betroffenen Knoten ermöglicht. Dokumente und Patienten werden dazu bei ihrer Anlage oder bei einem Verschiebevorgang an einem zentralen Suchdienst unter Angabe einiger suchrelevanter Metadaten entfernt referenzierbar registriert. Dadurch kann die Suche über einen zentralen Dienst durchgeführt werden. Der zentrale Dienst leitet die konkreten Anfragen nach Dokumenten dann an den datenhaltenden Knoten weiter.

Beziehungen - Grafik




Beziehungen - Detail

- **Single Storage-Point per Patient -> Zentraler Suchdienst** :
  Die Verwendung des Single-Storage-Point-per-Patient-Patterns vereinfacht den Aufbau eines zentralen Suchdienstes. Durch die eindeutige Zuordnung zwischen Patient und speicherndem Systemknoten kann dieser bereits zu Beginn einer Suchanfrage eindeutig identifiziert werden. Das ermöglicht ein eindeutiges Routing der Suchanfrage an einen jeweils einzelnen zuständigen Systemknoten. Dadurch wird eine zum Teil unnötige Belastung der anderen Systemknoten bei Suchanfragen vermieden. Ein weiterer Vorteil dieser Kombination liegt darin, dass im zentralen Suchdienst lediglich ein Index über die Zuordnung zwischen Knoten und Patient und nicht über alle verfügbaren Dokumente aufgebaut und vorgehalten werden muss.
- **Fully Distributed Patient Documents -> Zentraler Suchdienst** : Findung von Dokumenten
  Um Dokumente in einem System aus einer Vielzahl von Datenquellen wieder finden zu können, kann ein zusätzlicher Knoten in das System eingefügt werden, der als zentraler Suchdienst agiert. Ein zentraler Suchdienst muss in diesem Fall entweder von den verteilten Knoten, die Dokumente speichern über die Entstehung neuer Dokumente informiert werden, oder diese in regelmäßigen Abständen nach neuen Dokumenten durchsuchen. Auf diese Weise kann ein Index über die im Gesamtsystem verfügbaren Dokumente erstellt werden, der für die Umsetzung eines zentralen Suchdiensts notwendig ist.

  Umgekehrt ist für die Kombination aus Fully Distributed Patient Documents und Zentraler Suchdienst folgendes festzuhalten:
  Wird für die Verteilung der Dokumentenablage das Pattern Fully Distributed Patient Documents gewählt, so eignet sich als Registry für den Suchindex das Document-

Registry-Pattern. Alternativ kann auch eine Registry, die nur die Knoten, in denen Dokumente zu einem Patienten vorliegen verwaltet, verwendet werden. Nur durch einen Index, der alle Senken von Dokumenten zu Patienten zuordnet, kann vermieden werden, dass jeder Suchvorgang über alle Systemknoten hinweg ausgeführt werden muss.

- **Zentraler Suchdienst -> Full Access with Errors** : Steuerung der Zugriffsberechtigung auf Suchergebnisse
  Die Verwendung eines zentralen Suchdienstes führt dazu, dass entweder eine Full-Access-with-Errors-Lösung in Verbindung mit dezentralen Autorisierungslösungen oder eine Limited-Access-Lösung mit zentralen Autorisierungslösungen denkbar ist.
- **Zentraler Suchdienst -> Document Registry** : Bei fully distributed Patient Documents
  Wird für die Verteilung der Dokumentenablage das Pattern Fully Distributed Patient Documents gewählt, so eignet sich als Registry für den Suchindex das Document-Registry-Pattern.
- **Zentraler Suchdienst -> Limited Access** : Steuerung der Zugriffsberechtigung auf Suchergebnisse
  Die Verwendung eines zentralen Suchdienstes führt dazu, dass entweder eine Full-Access-with-Errors-Lösung in Verbindung mit dezentralen Autorisierungslösungen oder eine Limited-Access-Lösung mit zentralen Autorisierungslösungen denkbar ist.
- **Verteilung der Dokumentenablage pro Patient -> Suche im verteilten Dokumentensystem** : Benötigt
  Geerbte Beziehung: fuehrt_zu_beziehung
  Wenn Dokumente verteilt abgelegt werden ist es notwendig sie wiederfinden zu können.
- **Suche im verteilten Dokumentensystem -> Patientenidentifikation** : Zur eindeutigen Zuordnung der Suchergebnisse
  Geerbte Beziehung: fuehrt_zu_beziehung
  Um in einem verteilten Dokumentensystem gespeicherte Dokumente zu einem Patienten zu suchen, ist es notwendig, dass der Patient über die Grenzen der einzelnen Subsystem-Knoten hinweg eindeutig identifizierbar bleibt. Besonders Systeme mit heterogenen Patienten-IDs benötigen einen Mechanismus zur subsystemübergreifenden Patientenidentifikation.

# Anhang B: Quellen zur Untersuchung in Kapitel 2.6

## *1. Ergebnisse zum Suchbegriff "care record"*

| | |
|---|---|
| Titel: | Herausforderungen bei der Umsetzung der elektronischen Patientenakte und Gesundheitskarte in Österreich |
| URL: | http://www.springerlink.com/index/Q542382884205150.pdf |
| Quellenhinweis: | KP Pfeiffer… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2009 - Springer |

| | |
|---|---|
| Titel: | Stand, Möglichkeiten und Grenzen der Telemedizin in Deutschland |
| URL: | http://www.springerlink.com/index/11121nk75w665044.pdf |
| Quellenhinweis: | R Klar… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2009 - Springer |
| Titel: | EPA-Modelle im Vergleich: openEHR, HL7 V3 Specs, EN/ISO 13606, CCR |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-01-Blobel.pdf |
| Quellenhinweis: | B Blobel - Telemedizinführer Deutschland, 2008 - telemed-berlin.de |
| Titel: | Wege zur elektronischen Patientenakte |
| URL: | http://www.springerlink.com/index/81L3541P42777QM6.pdf |
| Quellenhinweis: | B Blobel… - Datenschutz und Datensicherheit-DuD, 2006 - Springer |
| Titel: | Aufbau einer einrichtungsübergreifenden Patientenakte in der Rhein-Neckar-Region |
| URL: | http://www.telemed-berlin.de/telemed2008/Medien/s152.pdf |
| Quellenhinweis: | O Heinze, A Brandner, R Brandner… - Telemed Tagungsband, …, 2008 - telemed-berlin.de |
| Titel: | Patientenzentrierte Dokumentation onkologischer Erkrankungen: Ein generisches XML-basiertes Informationsmodell zur syntaktischen und semantischen … |
| URL: | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2003/3818/pdf/Zusammenfassung_Astrid_Corinna_Wolff.pdf |
| Quellenhinweis: | A Wolff - Doktorarbeit, Ruprecht-Karls-Universität …, 2002 - archiv.ub.uni-heidelberg.de |
| Titel: | Individualisierte Gesundheitsportale als zentrale IT-Komponenten für Ambient Assisted Living: Motivation und Anforderungen |
| URL: | http://www.vde-verlag.de/proceedings-en/453138073.html |
| Quellenhinweis: | P Knaup-Gregori, M Ecker, F Albashiti… - … Assisted Living-AAL, 2009 - vde-verlag.de |
| Titel: | Die digitale Krankenakte als ethische Verpflichtung |
| URL: | http://www.cmsconnect.de/cokoehler/images/wiesloch-2002.pdf |
| Quellenhinweis: | CO Köhler - Die digitale Krankenakte im medizinischen Alltag, 2002 - cmsconnect.de |
| Titel: | Interoperabilität für Informationssysteme im Gesundheitswesen auf Basis medizinischer Standards |
| URL: | http://www.springerlink.com/index/JCJ7BW5Y8137KGRG.pdf |
| Quellenhinweis: | S Pedersen… - Informatik-Forschung und Entwicklung, 2004 - Springer |
| Titel: | Herausforderungen bei der Umsetzung der elektronischen Patientenakte und Gesundheitskarte in Österreich= Challenges in the implementation of electronic health … |
| URL: | http://cat.inist.fr/?aModele=afficheN&cpsidt=21288514 |
| Quellenhinweis: | KP Pfeiffer… - 2009 - cat.inist.fr |
| Titel: | Einleitung: Ausgangslage |
| URL: | http://www.ztg-nrw.de/ZTG/content/e35/e2990/e7026/lecture_downloads7623/object7630/huebner_flemming_ger.pdf |
| Quellenhinweis: | Z zu Daten der eGK - ztg-nrw.de |
| Titel: | Die Rolle des EHR bei der Modernisierung des Gesundheitswesens in Deutschland |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=E0aOHTXT5BQC&oi=fnd&pg=PA343&dq=%22care+record%22&ots=iGBforDxgS&sig=Q1neVB4mkhLDU33BVIZCKbLk_M4 |

| | |
|---|---|
| Quellenhinweis: | B Blobel… - … engineering and bioinformatics to the edge …, 2008 - books.google.com |
| Titel: | Literaturrecherche: Patient Health Record (PHR)-Die web-basierte Patienten-Akte |
| URL: | http://content.grin.com/document/v81357.pdf |
| Quellenhinweis: | CE Gerich - 2006 - GRIN Verlag |
| Titel: | SEMANTISCHE INTEROPERABILITÄT IM ELEKTRONISCHEN GESUNDHEITSDATENAUSTAUSCH MITTELS DUALER MODELLIERUNG-DER HL7 … |
| URL: | http://www.ehealth2011.at/archiv/eHealth2008/program/presentations/Postersession/bointner_paper.pdf |
| Quellenhinweis: | K Bointner… - ehealth2011.at |
| Titel: | CSC VITAE Suite (TM)-Ein Anwendungsbeispiel aus der Haus-und Heimpflege in Skandinavien |
| URL: | http://www.vde-verlag.de/proceedings-en/453323077.html |
| Quellenhinweis: | M Ernst, CP Kunze… - Ambient Assisted Living-AAL, 2011 - vde-verlag.de |
| Titel: | Zur Historie des Mutterpasses und seines Aktualisierungsbedarfs History of the German Antenatal Record (“Mutterpass”) and its Need for Updating |
| URL: | http://www.thieme-connect.com/ejournals/abstract/zgn/doi/10.1055/s-0029-1202786 |
| Quellenhinweis: | S Schling, P Hillemanns… - Z Geburtshilfe Neonatol, 2009 - thieme-connect.com |
| Titel: | Shared Care, Telemedizin und elektronische Krankenakte-Drei Pfeiler ei-nes neuen Gesundheitswesens |
| URL: | http://www.informierung.de/cokoehler/Kunath60.pdf |
| Quellenhinweis: | CO Köhler - informierung.de |
| Titel: | Nutzen und Kosten der Elektronischen Patientenakte in Krankenhäusern |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-16-Uslu.pdf |
| Quellenhinweis: | AM Uslu - telemed-berlin.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:EO0o9Rhp0qAJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | J Heart - Circulation, 2006 |
| Titel: | Der ePflegebericht: Aktueller Stand und |
| URL: | http://www1.messe-berlin.de/vip8_1/website/MesseBerlin/htdocs/Bilder_upload/Event-Datenbank/2022.PDF |
| Quellenhinweis: | D Flemming, U Hübner… - messe-berlin.de |
| Titel: | Web-basierte multi-methodische Evaluation Methodologie und Technische Umsetzung für ein web-basiertes Informationssystem über Krankenhäuser |
| URL: | http://scout.imib.rwth-aachen.de:8080/isg-med/pdfs/wessel2006_gmds_scout_pres.pdf |
| Quellenhinweis: | C Weßel, F Weymann… - scout.imib.rwth-aachen.de |
| Titel: | Kongress Medizin und Gesellschaft 2007 |
| URL: | http://www.egms.de/en/meetings/gmds2007/07gmds498.shtml |
| Quellenhinweis: | K Denecke - egms.de |
| Titel: | „EPA. nrw “, ein Projekt der Landesregierung NRW mit Partnern aus Industrie und Selbst-verwaltung |
| URL: | http://www.telemedizinf%C3%BChrer.de/free/2008/kuehn_75_80.pdf |
| Quellenhinweis: | S Kühn - Jäckel, A.(Hg.): Telemedizinführer …, 2008 - xn--telemedizinfhrer-uzb.de |

| | |
|---|---|
| Titel: | e-Health |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=XgEJwTpaNn8C&oi=fnd&pg=PR15&dq=%22care+record%22&ots=_lkiP9ryK9&sig=TbQyT-YIfZO3an_O-CtXQfBx0UY |
| Quellenhinweis: | K Jähn… - 2003 - books.google.com |
| Titel: | Einsicht in die Krankenakte. |
| URL: | http://iig.umit.at/dokumente/bsc_freina.pdf |
| Quellenhinweis: | K Freina - Studie über Akzeptanz, Nutzung und Nützlichkeit aus …, 2006 - iig.umit.at |
| Titel: | Europäische und internationale Perspektiven von Telematik im Gesundheitswesen |
| URL: | http://ehealth.gvg-koeln.de/cms/medium/699/EntwurfGVGStudie010215.pdf |
| Quellenhinweis: | S Schug - Schriftenreihe der GVG. Bd - ehealth.gvg-koeln.de |
| Titel: | Grundlagen der Qualitätszertifizierung von Diensten im Rahmen der Elektronischen Gesundheitsakte |
| URL: | http://iig.umit.at/dokumente/phd_hoerbst.pdf |
| Quellenhinweis: | A Hörbst - Institute for Health Information Systems, 2008 - iig.umit.at |
| Titel: | Systematische Analyse und Bewertung transmuraler Schnittstellen der Tiroler e-Health-Aktivitäten anhand des IHE IT-I-Frameworks |
| URL: | http://iig.umit.at/dokumente/msc_stark.pdf |
| Quellenhinweis: | ME des Akademischen Grades… - iig.umit.at |
| Titel: | Informationsintegration in Gesundheitsversorgungsnetzen |
| URL: | http://www.springerlink.com/index/GL0502134X466236.pdf |
| Quellenhinweis: | R Lenz, M Beyer, C Meiler, S Jablonski… - Informatik-Spektrum, 2005 - Springer |
| Titel: | Entwicklung eines normierten Pflegeentlassungsberichtes auf Basis der Clinical Document Architecture 2.0: Technische Umsetzung |
| URL: | http://www.meduniwien.ac.at/msi/mias/studarbeiten/2007-DA-Wagner.pdf |
| Quellenhinweis: | W Patrick… - meduniwien.ac.at |
| Titel: | Internationale Entwicklung des elektronischen lebensbegleitenden Gesundheitsakts am Beispiel Kanada und USA |
| URL: | http://www.meduniwien.ac.at/msi/mias/studarbeiten/2006-DA-Egelkraut.pdf |
| Quellenhinweis: | R Egelkraut… - meduniwien.ac.at |
| Titel: | Podiumsdiskussion Gesellschaftliche Aspekte, Sicherheit und Datenschutz |
| URL: | http://www.springerlink.com/index/gn67w62686438h11.pdf |
| Quellenhinweis: | MT Tinnefeld - eHealth: Innovations-und Wachstumsmotor für Europa, 2006 - Springer |
| Titel: | Rechtliche und praktische Probleme der Integration von Telemedizin in das Gesundheitswesen in Deutschland |
| URL: | http://deposit.ddb.de/cgi-bin/dokserv?idn=959449531&dok_var=d1&dok_ext=pdf&filename=959449531.pdf |
| Quellenhinweis: | C Dierks - 1999 - deposit.ddb.de |
| Titel: | Analyse von Werkzeugen zur CDA-Erstellung |
| URL: | http://www.cs.uni-goettingen.de/Filepool/Theses/gaug-zfi-bm-2005-31.pdf |
| Quellenhinweis: | M Kaspar - Zentrum für Informatik, 2005 - cs.uni-goettingen.de |
| Titel: | Telekommunikation, eCommunication |
| URL: | http://www.springerlink.com/index/p446p752523565p7.pdf |
| Quellenhinweis: | M wie Röntgenbilder… - Gesundheitstelematik, 2006 - Springer |
| Titel: | Integration genomischer Daten in Systeme der klinischen Versorgung und der klinischen |

| | |
|---|---|
| | Forschung |
| URL: | http://www.cs.uni-goettingen.de/Filepool/Theses/gaug-zfi-msc-2008-03.pdf |
| Quellenhinweis: | M Kaspar - 2008 - cs.uni-goettingen.de |
| Titel: | Konzepte zur elektronischen Arztbriefschreibung und–Übermittlung |
| URL: | http://miami.uni-muenster.de/servlets/DerivateServlet/Derivate-4823/diss_butta.pdf |
| Quellenhinweis: | K im Gesundheitswesen - miami.uni-muenster.de |
| Titel: | Erstellung von XML-Signaturen für Dokumente nach Clinical Documents Architecture–R2 |
| URL: | http://aerztekamm-rheinland.de/downloads/aekno/2008-05-20-CDA2_Signatur_Spezifikation_V1_4_final.pdf |
| Quellenhinweis: | BÄNÄ Westfalen-Lippe - 2007 - aerztekamm-rheinland.de |
| Titel: | Berichter: Herr Universitätsprofessor Dr. med. Dr. rer. nat. Klaus Spitzer Herr Universitätsprofessor Dr. rer. nat. Klaus Willmes-von Hinkeldey |
| URL: | http://deposit.ddb.de/cgi-bin/dokserv?idn=997151188&dok_var=d1&dok_ext=pdf&filename=997151188.pdf |
| Quellenhinweis: | J Rams - deposit.ddb.de |

## *2. Ergebnisse zum Suchbegriff "distributed health record"*

| | |
|---|---|
| Titel: | Security requirements and solutions in distributed electronic health records |
| URL: | http://www.ingentaconnect.com/content/els/01674048/1997/00000016/00000003/art84526 |
| Quellenhinweis: | B Blobel - Computers and Security, 1997 - ingentaconnect.com |
| Titel: | An eConsent-based system architecture supporting cooperation in integrated healthcare networks |
| URL: | http://iospress.metapress.com/index/7e64lx91hl1e9c5v.pdf |
| Quellenhinweis: | J Bergmann, OJ Bott, I Hoffmann… - Studies in health …, 2005 - IOS Press |
| Titel: | Policy contexts: Controlling information flow in parameterised RBAC |
| URL: | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1206964 |
| Quellenhinweis: | A Belokosztolszki, DM Eyers… - Policies for Distributed …, 2003 - ieeexplore.ieee.org |
| Titel: | Consumer-centric and privacy-preserving identity management for distributed e-health systems |
| URL: | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4438938 |
| Quellenhinweis: | R Au… - … on System Sciences, Proceedings of the 41st …, 2008 - ieeexplore.ieee.org |
| Titel: | Security analysis and design based on a general conceptual security model and UML |
| URL: | http://www.springerlink.com/index/g537g115051x671u.pdf |
| Quellenhinweis: | B Blobel, P Pharow… - High-Performance Computing and …, 1999 - Springer |
| Titel: | The Health Record: Kernel of a Medical Memory |
| URL: | http://rosselle1.unblog.fr/files/2008/03/articlerech.pdf |
| Quellenhinweis: | S Bringay, C Barry… - Proceeding of the Workshop …, 2004 - rosselle1.unblog.fr |
| Titel: | Comprehensive Health Records Based on Advanced Card Technology and Networked Computers: Are They Feasible? |
| URL: | http://books.google.de/books?hl=en&lr=&id=dDEwACNdgxwC&oi=fnd&pg=PA158&d |

| | |
|---|---|
| | q=%22distributed+health+record%22&ots=l_t0MeXy6v&sig=oYHmRYaN9wa5o0XpmZE5be0XF9o |
| Quellenhinweis: | JR Mohr - Health Cards95: Proceedings of the Health Cards' 95 …, 1995 - books.google.com |
| Titel: | Model-Based Security Analysis of Health Care Networks |
| URL: | http://qe-informatik.uibk.ac.at/%7Efrank/pdfs/bims08.pdf |
| Quellenhinweis: | R Breu, F Innerhofer-Oberperfler… - Proc. eHealth2008, …, 2008 - qe-informatik.uibk.ac.at |
| Titel: | Privacy enforcement for distributed healthcare queries |
| URL: | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5191221 |
| Quellenhinweis: | M Siegenthaler… - … Computing Technologies for …, 2009 - ieeexplore.ieee.org |
| Titel: | Medical and Home automation Sensor Networks for Senior Citizens Telehomecare |
| URL: | http://hal.archives-ouvertes.fr/inria-00431096/ |
| Quellenhinweis: | S Nourizadeh, C Deroussent, YQ Song… - 2009 - hal.archives-ouvertes.fr |
| Titel: | Ontology-based distributed health record management system |
| URL: | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4648393 |
| Quellenhinweis: | C Cenan, G Sebestyen, G Saplacan… - … and Processing, 2008 …, 2008 - ieeexplore.ieee.org |
| Titel: | Universal Electronic Health Record MUDR |
| URL: | http://iospress.metapress.com/index/J8Y28CURGNNQW9G3.pdf |
| Quellenhinweis: | M Duplaga - IOS Press |
| Titel: | E-Health and Telemedicine |
| URL: | http://www.springerlink.com/index/4R32146W4WU18870.pdf |
| Quellenhinweis: | G Graschew, TA Roelofs, S Rakowsky… - International Journal of …, 2006 - Springer |
| Titel: | Content-based medical image retrieval in peer-to-peer systems |
| URL: | http://portal.acm.org/citation.cfm?id=1883103 |
| Quellenhinweis: | A Charisi… - … of the 1st ACM International Health …, 2010 - portal.acm.org |
| Titel: | Annex 1-“Description of Work” Detailed JPA for the period January 2006–June 2007 |
| URL: | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.103.4677&rep=rep1&type=pdf |
| Quellenhinweis: | NOF EXCELLENCE - Citeseer |

## *3. Ergebnisse zum Suchbegriff "elektronische Fallakte"*

| | |
|---|---|
| Titel: | Keiner angegeben |
| URL: | http://scholar.google.de/scholar?cites=4190385115442555740&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | IU elektronische Fallakte - 2006 |
| Titel: | Die elektronische Gesundheitskarte |
| URL: | https://www.datenschutzzentrum.de/medizin/gesundheitskarte/dud_gesundheitskarte.pdf |
| Quellenhinweis: | T Weichert - Datenschutz und Datensicherheit, 2004 - datenschutzzentrum.de |
| Titel: | eHealth verspricht individuelle und bedürfnis-gerechte Betreuung der Patienten-Perspektive zur Steigerung von Qualität und Effizienz |

| | |
|---|---|
| URL: | http://www.thieme-connect.com/ejournals/abstract/klinikarzt/doi/10.1055/s-2007-970251 |
| Quellenhinweis: | P profitieren von der elektronischen Fallakte - Klinikarzt, 2007 - thieme-connect.com |
| Titel: | eFA-Die Elektronische Fallakte als Basis für sektorübergreifende Prozesse |
| URL: | http://en.scientificcommons.org/28076252 |
| Quellenhinweis: | J Neuhaus… - 2007 - en.scientificcommons.org |
| Titel: | Die elektronische Fallakte-ein Standard für die einrichtungsübergreifende Kommunikation |
| URL: | http://en.scientificcommons.org/45722961 |
| Quellenhinweis: | C Reuter, J Neuhaus, J Caumanns… - 2009 - en.scientificcommons.org |
| Titel: | |
| URL: | http://direct.bl.uk/research/25/35/RN194968587.html?source=googlescholar |
| Quellenhinweis: | HE Kruger-Brand - DEUTSCHES ARZTEBLATT …, 2006 - … DER DEUTSCHER AERZTE- … |
| Titel: | |
| URL: | http://scholar.google.de/scholar.bib?q=info:_WOI0tLONsUJ:scholar.google.com/&output=citation&hl=en&as_sdt=0,5&ct=citation&cd=6 |
| Quellenhinweis: | HE Kruger-Brand - … Mitteilungen-Ausgabe A, 2006 - Koln: Deutscher Arzte-Verlag GmbH, … |
| Titel: | Föderatives Identitätsmanagement am Beispiel der elektronischen Fallakte |
| URL: | http://www.springerlink.com/index/L4518084790K1V73.pdf |
| Quellenhinweis: | O Boehm… - Informatik-Spektrum, 2007 - Springer |
| Titel: | Elektronische Fallakten zur einrichtungsübergreifenden Kooperation |
| URL: | http://en.scientificcommons.org/26947842 |
| Quellenhinweis: | J Caumanns, O Boehm… - 2007 - en.scientificcommons.org |
| Titel: | Elektronische Fallakten zur sicheren einrichtungsübergreifenden Kommunikation |
| URL: | http://www.e-fallakte.de/Elektronische_FallAkten_zur_sicheren_einrichtungs%C3%BCbergreifenden_Kooperation.pdf |
| Quellenhinweis: | J Caumanns, O Boehm… - 2007 - e-fallakte.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:rnzxF1pipyIJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | ABU Erlangen, ABU Erlangen, A Mrosek… |
| Titel: | Realisierung komplexer Sicherheitsdienste unter Verwendung der Standards WS-Trust und SAML am Beispiel der elektronischen Fallakte |
| URL: | http://www.fallakte.de/download/diplomarbeiten/diplomarbeit_rkuhlisch.pdf |
| Quellenhinweis: | R Kuhlisch… - fallakte.de |
| Titel: | Deklarative Sicherheit zur Spezifikation und Implementierung der elektronischen Fallakte |
| URL: | http://subs.emis.de/LNI/Proceedings/Proceedings174/P-174.pdf#page=40 |
| Quellenhinweis: | R Kuhlisch… - … für Informatik (GI) publishes this series in …, 2010 - subs.emis.de |
| Titel: | 54. Jahrestagung der Deutschen Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie eV (GMDS) |
| URL: | http://www.egms.de/en/meetings/gmds2009/09gmds308.shtml |
| Quellenhinweis: | H Kirchner, HU Prokosch, J Dudeck, KH Jöckel… - egms.de |

| | |
|---|---|
| Titel: | Trends und Entwicklungen der Krankenhaus-IT-Technologie |
| URL: | http://www.springerlink.com/index/GJ3776652Q221L1P.pdf |
| Quellenhinweis: | M Staemmler - Steuerung der IT im Klinikmanagement, 2010 - Springer |
| Titel: | Kardiologische Betreuungsmodelle |
| URL: | http://www.springerlink.com/index/l6p27938527q6718.pdf |
| Quellenhinweis: | A Müller, J Schweizer… - Prävention und Gesundheitsförderung, 2008 - Springer |
| Titel: | eHealth als Katalysator des Wandels im Kliniksektor |
| URL: | http://www.klinikum.uni-muenchen.de/Medizinisch-Administrative-Informationstechnologie/download/inhalt/veroeffentlichungen/eHealth-katalysator-wandel-kliniksektor.pdf |
| Quellenhinweis: | W Swoboda, S Villain… - Klusen N/Meusch A …, 2008 - klinikum.uni-muenchen.de |
| Titel: | D2D-MammaAkte der KV Nordrhein |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=qMv8BxX7MewC&oi=fnd&pg=PA263&dq=%22elektronische+Fallakte%22&ots=yppTe8xUlI&sig=ohfXJyHAzfYnMQdOn3uuZRQSgbo |
| Quellenhinweis: | G Mohr, E Gehlen… - 54. Kongress der …, 2003 - books.google.com |
| Titel: | Generischer Architekturansatz für Telemedizin Portale und verteilte Krankenakten |
| URL: | http://epub.uni-regensburg.de/8909/1/s065.pdf |
| Quellenhinweis: | W Wiedermann, T Athanassios… - 2008 - epub.uni-regensburg.de |
| Titel: | „Telemedizin “im Spiegel des Deutschen Ärzteblattes. Problemfelder und Akzeptanzfaktoren |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=SSg0St9ORlkC&oi=fnd&pg=PA51&dq=%22elektronische+Fallakte%22&ots=g6huv63AJt&sig=xc99Z9v4oR40zViFiIHk7zxZBiE |
| Quellenhinweis: | S Kreucher, D Groß… - Akzeptanz, Nutzungsbarrieren und … - books.google.com |
| Titel: | Hintergrund und Fragestellung |
| URL: | http://www.site-telemed.de/download/mueller_helms.pdf |
| Quellenhinweis: | A Müller, J Schweizer… - 2008 - site-telemed.de |
| Titel: | Jubiläum für Datenschutz und Informationsfreiheit |
| URL: | http://www.springerlink.com/index/Y54250J205617U67.pdf |
| Quellenhinweis: | A Dix - Datenschutz und Datensicherheit-DuD, 2010 - Springer |
| Titel: | Auf dem Weg zu einem prosperierenden europäischen eHealth-Markt–Hindernisse ohne Ausweg? |
| URL: | http://www.telemedizinf%C3%BChrer.de/free/2008/reiher_292_296.pdf |
| Quellenhinweis: | M Reiher, F Oemig… - xn--telemedizinfhrer-uzb.de |
| Titel: | Nutzen und Kosten der Elektronischen Patientenakte in Krankenhäusern |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-16-Uslu.pdf |
| Quellenhinweis: | AM Uslu - telemed-berlin.de |
| Titel: | Authentisierung/Authentifizierung |
| URL: | http://www.fallakte.de/download/eFA1.2-Sicherheit_und_Datenschutz/eFA1.2-Authentication_v1.2.0.02.pdf |
| Quellenhinweis: | Ü über die Client… - 2008 - fallakte.de |
| Titel: | Elektronische Gesundheitskarte Elektronische Patientenakte |
| URL: | http://www.aegmd.de/files/lehrbrief_4_2009.pdf |

| | |
|---|---|
| Quellenhinweis: | WM Lamers… - 2009 - aegmd.de |
| Titel: | Die unerträgliche Geschichte der GesundheitskarteinDeutschland |
| URL: | http://www.nachdenkseiten.de/upload/pdf/geschichte_der_e-card_Prof_G_Schweim_GMS.pdf |
| Quellenhinweis: | HG Schweim - 2007 - nachdenkseiten.de |
| Titel: | Die unerträgliche Geschichte der Gesundheitskarte in Deutschland |
| URL: | http://www.egms.de/en/journals/mibe/2007-3/mibe000052.shtml |
| Quellenhinweis: | D Fakten - egms.de |
| Titel: | IT-Beratung im Gesundheitssystem aus systemtheoretischer Perspektive |
| URL: | http://content.grin.com/document/v141984.pdf |
| Quellenhinweis: | CH Timwo Monthe - 2009 - GRIN Verlag |
| Titel: | Mögliche Ansätze für eine gute Transition |
| URL: | http://www.springerlink.com/index/D56XN84R37307825.pdf |
| Quellenhinweis: | M Haubitz… - Der Nephrologe, 2011 - Springer |
| Titel: | Die betriebswirtschaftliche Bewertung der IT-Performance im Krankenhaus am Beispiel eines Benchmarking-Projekts |
| URL: | http://www.springerlink.com/index/R253252G61K05107.pdf |
| Quellenhinweis: | A Simon - Steuerung der IT im Klinikmanagement, 2010 - Springer |
| Titel: | It-beratung im Gesundheitssystem aus systemtheoretischer Perspektive:-Status quo |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=GHz_NG97lq0C&oi=fnd&pg=PT13&dq=%22elektronische+Fallakte%22&ots=6DiJu8eq6w&sig=um2cefWcYvQaSeqcK-L4Tmhbvb8 |
| Quellenhinweis: | MSCHT Monthe - 2009 - books.google.com |
| Titel: | Der Patient und seine Akte: elektronische Patientenakten und das Selbstbestimmungsrecht |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=ZO7qCIDEyeIC&oi=fnd&pg=PA1&dq=%22elektronische+Fallakte%22&ots=i1t9aV4f9W&sig=8e20ZqpKTmAXjxBaib8CoujrKvo |
| Quellenhinweis: | S Lütkehaus - 2010 - books.google.com |
| Titel: | Kollege Computer: Moderne Medizin durch Telematik |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=3tSltT1_ckoC&oi=fnd&pg=PA1&dq=%22elektronische+Fallakte%22&ots=L-UivJcaOc&sig=WMGPuPfyOLvff0WPzBwclsHSZFw |
| Quellenhinweis: | H Lohmann… - 2009 - books.google.com |
| Titel: | RFIDs im Krankenhaus |
| URL: | http://www.net2.uni-tuebingen.de/fileadmin/RI/teaching/seminar_medizin/ss06/vortraege/muelling.pdf |
| Quellenhinweis: | K Mülling - Online im Internet: http://net. informatik. …, 2006 - net2.uni-tuebingen.de |
| Titel: | Integrierte und hybride Konstruktion von Software-Produktlinien |
| URL: | http://deposit.ddb.de/cgi-bin/dokserv?idn=1007551526&dok_var=d1&dok_ext=pdf&filename=1007551526.pdf |
| Quellenhinweis: | DIU Dinger - deposit.ddb.de |
| Titel: | Rahmenarchitektur und technischer Lösungsvorschlag zur Umsetzung eines bürgerfreundlichen Identitätsmanagements in Verwaltung und Wirtschaft |
| URL: | http://isprat.net/fileadmin/downloads/pdfs/Buergerfreundliches_Identitaetsmanagement_- |

| | |
|---|---|
| | _Rahmenarchitektur_und_technischer_Loesungsvorschlag.pdf |
| Quellenhinweis: | J Fromm, P Hoepner, A Steinacker… - 2009 - isprat.net |
| Titel: | Regionale Verbünde in der Onkologie |
| URL: | http://www.springerlink.com/index/h46645862627875j.pdf |
| Quellenhinweis: | M Zünkeler… - Der Onkologe, 2009 - Springer |
| Titel: | Forschungsorientierte medizinische Langzeitdokumentation auf Basis der gematik-Infrastruktur |
| URL: | http://www.cs.uni-goettingen.de/Filepool/Theses/gaug-zfi-bm-2007-15.pdf |
| Quellenhinweis: | K Helbing - 2007 - cs.uni-goettingen.de |
| Titel: | Aufbau eines strategisch ausgerichteten sowie erfolgsorien-tierten Vergütungssystems für Integrierte Versorgungsformen |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=OqPfnGH0ytQC&oi=fnd&pg=PA249&dq=%22elektronische+Fallakte%22&ots=K-XVYmd-O1&sig=3iN_DU9WVZLDury0ASVbiYlUw5o |
| Quellenhinweis: | M Gröbner… - Innovative Versorgungsformen im …, 2009 - books.google.com |
| Titel: | Betriebliches Informationswesen |
| URL: | http://www.springerlink.com/index/L2P78212L1621766.pdf |
| Quellenhinweis: | A Frodl - Gesundheitsbetriebslehre, 2010 - Springer |
| Titel: | Akzeptanz, Nutzungsbarrieren und ethische Implikationen neuer Medizintechnologien |
| URL: | http://www.upress.uni-kassel.de/online/frei/978-3-89958-930-6.volltext.frei.pdf |
| Quellenhinweis: | D Groß, G Gründer… - upress.uni-kassel.de |
| Titel: | Marketing ist Chefsache–auch in der Abteilungsführung |
| URL: | http://posterous.com/getfile/files.posterous.com/krojer/9FOnPJ5iYyDCbPEucJVFVZCBdxCgnNyLAHwzY8zaBlV5vVyGsUBmYF9pQaxb/Buchkapitel_Marketing_Klinik_A.pdf |
| Quellenhinweis: | DW Bienert - posterous.com |
| Titel: | Moderne Dienstleistungen am Arbeitsmarkt |
| URL: | http://www.cgb-nrw.de/dateien/presseinfos/Hartz_Bericht.pdf |
| Quellenhinweis: | N Bensel, J Fiedler, H Fischer, P Gasse… - … der Kommission zum …, 2002 - cgb-nrw.de |
| Titel: | Case Management im Übergangsmanagement von der Schule in den Beruf am Beispiel des Projektes" Kompetenzagenturen" |
| URL: | http://content.grin.com/document/v167050.pdf |
| Quellenhinweis: | J Mahlstaedt - 2011 - GRIN Verlag |

## *4. Ergebnisse zum Suchbegriff "elektronische Krankenakte"*

| | |
|---|---|
| Titel: | KAS, KIS, EKA, EPA, EGA, E-Health:--Ein Pladoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik |
| URL: | http://www.imi.med.uni-erlangen.de/fileadmin/alt/team/download/mis_begriffsdefinitionen.pdf |
| Quellenhinweis: | HU Prokosch - Informatik Biometrie und Epidemiologie in …, 2001 - imi.med.uni-erlangen.de |

| | |
|---|---|
| Titel: | Psychiatrie und Internet: Möglichkeiten, Risiken, Perspektiven |
| URL: | http://www.springerlink.com/index/7JDBLVYFH49T26LM.pdf |
| Quellenhinweis: | U Hegerl… - Der Nervenarzt, 2002 - Springer |
| Titel: | Elektronische oder papiergebundene Patientenakte Ein Kosten-Nutzen-Vergleich |
| URL: | http://www.springerlink.com/index/LHQHJGX3M7UNW4C2.pdf |
| Quellenhinweis: | AS Neubauer, S Priglinger… - Der Ophthalmologe, 2001 - Springer |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=6304160351938917 42&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | D Hölzel, K Adelhard, R Eckel, A König… - 1994 - Ecomed |
| Titel: | Medizinische Informationssysteme und Elektronische Krankenakten |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=XfhnwGCS9pYC&oi=fnd&pg=PA1&dq=%22elektronische+Krankenakte%22&ots=COSLWTvt5J&sig=qEpSHpumJBuPl_49bfE68q1XNFY |
| Quellenhinweis: | P Haas - 2004 - books.google.com |
| Titel: | Sprechen Sie LOINC |
| URL: | http://www.loinc.de/literatur/Einfuehrung_in_LOINC_%28HL7News8-2000%29.pdf |
| Quellenhinweis: | C McDonald, G Schadow, J Suico… - HL7-Mitteilungen, 2000 - loinc.de |
| Titel: | Unterstützung medizinischer Entscheidungen durch eine elektronische Krankenakte |
| URL: | http://www.reference-global.com/doi/abs/10.1515/bmte.1993.38.s1.225 |
| Quellenhinweis: | K Adelhard, R Eckel, D Hölzel… - Biomedizinische …, 1993 - reference-global.com |
| Titel: | Die elektronische Krankenakte-Eine Sicherheitsstrategie |
| URL: | http://sit.sit.fraunhofer.de/smv/publications/download/DACH08-rieke-v2.pdf |
| Quellenhinweis: | P Ochsenschläger, R Rieke… - DACH Security, 2008 - sit.sit.fraunhofer.de |
| Titel: | Die Inzidenz von Zwangsmassnahmen als Qualitätsindikator in psychiatrischen Kliniken. Probleme der Datenerfassung und–verarbeitung und erste Ergebnisse |
| URL: | http://www.thieme-connect.com/ejournals/abstract/psychiat-praxis/doi/10.1055/s-2005-866920 |
| Quellenhinweis: | V Martin, W Kuster, M Baur, U Bohnet… - Psychiat …, 2007 - thieme-connect.com |
| Titel: | hightech im Notfall |
| URL: | http://www.springerlink.com/index/5MTLFTKK82Q6F95Y.pdf |
| Quellenhinweis: | P Herrmann - Notfall & Rettungsmedizin, 1997 - Springer |
| Titel: | Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2006-926810 |
| Quellenhinweis: | C Jürgens, S Antal, F Heydenreich… - Klin Monatsbl …, 2006 - thieme-connect.com |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=18058433271253549118&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | L Zachewitz… - Interdisziplinärer Workshop KIS/RIS/PACS, Schloss …, 2004 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=661184500232075165&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Dietel… - Zeitschrift fur Arztliche Fortbildung und …, 2001 |

| | |
|---|---|
| Titel: | Die digitale Krankenakte als ethische Verpflichtung |
| URL: | http://www.cmsconnect.de/cokoehler/images/wiesloch-2002.pdf |
| Quellenhinweis: | CO Köhler - Die digitale Krankenakte im medizinischen Alltag, 2002 - cmsconnect.de |
| Titel: | Aspekte der elektronischen Krankenakte in der Radiologie |
| URL: | http://www.springerlink.com/index/HHR6HU1BRQED1YQ7.pdf |
| Quellenhinweis: | K Adelhard, S Nissen-Meyer… - Der Radiologe, 1999 - Springer |
| Titel: | „Einsatz eines mobilen Computersystems zur Befunderfassung in der Zahnheilkunde-eine Multicenterstudie" |
| URL: | http://www.medizin.uni-koeln.de/projekte/gmds-mocomed/workshop2001/tagungsband/9.pdf |
| Quellenhinweis: | M Lange, R Ide… - 2001):„Mobiles Computing in der …, 2001 - medizin.uni-koeln.de |
| Titel: | Prozessoptimierung durch moderne Krankenhaus-Informations-und Workflowsysteme |
| URL: | http://www.springerlink.com/index/jx507676157067tm.pdf |
| Quellenhinweis: | HU Prokosch - eHealth: Innovations-und Wachstumsmotor für Europa, 2006 - Springer |
| Titel: | Verteilte Architekturen zur intra-und inter-institutionellen Integration von Patientendaten |
| URL: | http://se.informatik.uni-oldenburg.de:30000/73/1/gmds2004.pdf |
| Quellenhinweis: | L Bischofs, W Hasselbring, H Niemann… - 2004 - se.informatik.uni-oldenburg.de |
| Titel: | Implementierung einer elektronischen Krankenakte |
| URL: | http://www.springerlink.com/index/V883TH4253748J5P.pdf |
| Quellenhinweis: | A Grüner, A Ljutow, W Schleinzer… - Der Schmerz, 2008 - Springer |
| Titel: | Einsicht in die Krankenakte. |
| URL: | http://iig.umit.at/dokumente/bsc_freina.pdf |
| Quellenhinweis: | K Freina - Studie über Akzeptanz, Nutzung und Nützlichkeit aus …, 2006 - iig.umit.at |
| Titel: | Lehren für eGK aus relevanten internationalen Strukturprojekten |
| URL: | http://ehealth.gvg-koeln.de/cms/medium/290/01_Blobel050503.pdf |
| Quellenhinweis: | B Blobel - 2005 - ehealth.gvg-koeln.de |
| Titel: | Integration von Anwendungssystemen des stationären und ambulanten Versorgungssektors am Beispiel des Projektes Mamma@ kte. nrw |
| URL: | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.3988&rep=rep1&type=pdf#page=42 |
| Quellenhinweis: | P Haas, M Eckenbach, H Plagge… - EAI 2005 Enterprise … - Citeseer |
| Titel: | Die Implementierung von bIT4health im Spiegel der europäischen Initiativen sowie der fortgeschrittenen Programme anderer Länder |
| URL: | http://www.charite.de/medinfo/Userpages/Mitarbeiter/Stiller/CD/2005/Telemed-2005-09-Blobel-Vortrag.pdf |
| Quellenhinweis: | B Blobel - Steyer, Günter und Tolxdorff, Thomas,(eds.) bit for bit- …, 2005 - charite.de |
| Titel: | Datenschutz in Krankenhausinformationssystemen |
| URL: | http://www.staff.uni-mainz.de/pommeren/Artikel/vis95.pdf |
| Quellenhinweis: | K Pommerening - … , HH; Gerhardt-Häckl, W.: Verläßliche IT- …, 1995 - staff.uni-mainz.de |
| Titel: | EPA-Modelle im Vergleich: openEHR, HL7 V3 Specs, EN/ISO 13606, CCR |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-01-Blobel.pdf |
| Quellenhinweis: | B Blobel - Telemedizinführer Deutschland, 2008 - telemed-berlin.de |
| Titel: | Standards für die medizinische Kommunikation und Dokumentation |

| | |
|---|---|
| URL: | http://se.informatik.uni-oldenburg.de:30000/100/1/TR-04-2003.pdf |
| Quellenhinweis: | S Pedersen… - 2003 - se.informatik.uni-oldenburg.de |
| Titel: | Ein graph-und objektorientiertes Patientendaten-Modell |
| URL: | http://www.staff.uni-mainz.de/pommeren/Artikel/patmod.pdf |
| Quellenhinweis: | R Müller… - 1994 - staff.uni-mainz.de |
| Titel: | Wege zur elektronischen Patientenakte |
| URL: | http://www.springerlink.com/index/81L3541P42777QM6.pdf |
| Quellenhinweis: | B Blobel… - Datenschutz und Datensicherheit-DuD, 2006 - Springer |
| Titel: | Sicheres und nachhaltiges eHealth |
| URL: | http://blog.eh-cc.de/wp-content/html/agdgi/down/KIS-2007_01_Blobel.pdf |
| Quellenhinweis: | B Blobel… - P. Schmücker, K.-H. Ellsässer (Hrsg.) - blog.eh-cc.de |
| Titel: | EDV-gestützte Qualitätssicherung in der pädiatrischen Diabetologie |
| URL: | http://www.mathematik.uni-ulm.de/sai/grabert/gmds97.htm |
| Quellenhinweis: | M Grabert, RW Holl, J Högel… - Medizinische …, 1997 - mathematik.uni-ulm.de |
| Titel: | Einführung in die Medizinische Informatik |
| URL: | http://www.umi.cs.tu-bs.de/emi/ws0708/emi_0708_Plan.pdf |
| Quellenhinweis: | R Haux, C Duvenkamp, N Gusew… - Signal, 2007 - umi.cs.tu-bs.de |
| Titel: | Elektronische Krankenakte |
| URL: | http://www.meb.uni-bonn.de/gmds/abstracts/0202i.html |
| Quellenhinweis: | K Hofmann, S Gräber… - meb.uni-bonn.de |
| Titel: | Webbasierte elektronische Krankenakte im Rahmen eines integrierten Versorgungskonzepts als Instrument der Qualitätssicherung Web-Based Electronic Patient … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-0028-1109193 |
| Quellenhinweis: | A Händel, AGM Jünemann… - Klin Monatsbl …, 2009 - thieme-connect.com |
| Titel: | Die elektronische Krankenakte: ein essentieller Teil der Telematikanwendungen im Gesundheitswesen |
| URL: | http://www.thieme-connect.com/ejournals/abstract/zblgyn/doi/10.1055/s-2000-10101 |
| Quellenhinweis: | R Engelbrecht - Zentralbl Gynakol, 2000 - thieme-connect.com |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:w52ovWjs6GAJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | S Müller, F Gerdsen, HU Prokosch… |
| Titel: | Shared Care, Telemedizin und elektronische Krankenakte-Drei Pfeiler ei-nes neuen Gesundheitswesens |
| URL: | http://www.informierung.de/cokoehler/Kunath60.pdf |
| Quellenhinweis: | CO Köhler - informierung.de |
| Titel: | Elektronische Krankenakte |
| URL: | http://www.meb.uni-bonn.de/gmds/abstracts/0396i.html |
| Quellenhinweis: | KH Ellsässer, M Gutsche… - meb.uni-bonn.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar.bib?q=info:bzKpllOByvMJ:scholar.google.com/&output=citation&hl=en&as_sdt=0,5&ct=citation&cd=37 |
| Quellenhinweis: | W Moser, T Diedrich… - KI, 1997 |

| | |
|---|---|
| Titel: | |
| URL: | http://direct.bl.uk/research/37/0D/RN224246636.html?source=googlescholar |
| Quellenhinweis: | A Gr ├╜ ner, A Ljutow... - SCHMERZ- ..., 2008 - Springer Science+ Business Media |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cluster=11274685247019809687&hl=en&as_sdt=0,5 |
| Quellenhinweis: | D Balthasar - 2003 - Universität Konstanz, … |
| Titel: | VERGLEICH ELGA KONFORMER EHEALTH SYSTEME FÜR DEN WIENER KRANKENANSTALTENVERBUND |
| URL: | http://www.meduniwien.ac.at/msi/mias/papers/Moosheer2010.pdf |
| Quellenhinweis: | J Moosheer, K Hölzl… - meduniwien.ac.at |
| Titel: | Informationsmanagement in Disease Management Programmen |
| URL: | http://www.springerlink.com/index/r55w31441387m477.pdf |
| Quellenhinweis: | J Ricke - eHealth: Innovations-und Wachstumsmotor für Europa, 2006 - Springer |
| Titel: | Krankenhausinformationssysteme |
| URL: | http://content.grin.com/document/v106541.pdf |
| Quellenhinweis: | S Knorr - 2002 - GRIN Verlag |
| Titel: | Was sind die aktuellen Aufgaben?-Medizinische Informatik in der Frauenheilkunde Medical Informatics in Obstetrics and Gynecology-The Actual Challenge? |
| URL: | http://www.thieme-connect.com/ejournals/abstract/zblgyn/doi/10.1055/s-2003-44577 |
| Quellenhinweis: | R Seufert, M Munz, H Pilch… - Zentralbl …, 2003 - thieme-connect.com |
| Titel: | Institut für Medizinische Informatik, Statistik und Epidemiologie |
| URL: | http://www.imise.uni-leipzig.de/Lehre/Semester/2008-09/MedDok/MedizinischeDokumentation.pdf |
| Quellenhinweis: | A Winter - imise.uni-leipzig.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:NGl1Vv067WYJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | P Knöll, B Krug, JBPPH Consulting… |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:ObcniYKBldIJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | C Dujat, A Häber… - … in Krankenhaus und …, 2005 - BoD–Books on Demand |
| Titel: | 49. Jahrestagung der Deutschen Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie (gmds) 19. Jahrestagung der Schweizerischen … |
| URL: | http://www.egms.de/en/meetings/gmds2004/04gmds039.shtml |
| Quellenhinweis: | I Castellanos, G Rellensmann… - egms.de |
| Titel: | Zukunftsweisende Kooperation von Wissenschaft und Wirtschaft-Telemedizinanwendungen in Forschung und Praxis |
| URL: | http://www.telemed-berlin.de/telemed2006/Beitraege/P06%20-%20BRINKMANN,%20LARS,%20Zukunftsweisende.pdf |
| Quellenhinweis: | L Brinkmann, M Spitzer, H Doan - telemed-berlin.de |
| Titel: | Darstellung des Programms nach Projektbereichen und Teilprojekten |
| URL: | http://www.sfb539.forschung.uni-erlangen.de/bereiche/C.5/Teilprojekt-C5-2003-04-09.pdf |

Quellenhinweis: I Vernetzung, AITS zur Qualitätssteigerung… - sfb539.forschung.uni-erlangen.de

## 5. Ergebnisse zum Suchbegriff "elektronische Patientenakte"

Titel:

URL: http://scholar.google.de/scholar?cites=15913506084618166625&as_sdt=2005&sciodt=0,5&hl=en

Quellenhinweis: P Schmücker, C Ohr, A Beß, HB Bludau, R Haux… - Informatik, Biometrie und …, 1998

Titel: Elektronische oder papiergebundene Patientenakte Ein Kosten-Nutzen-Vergleich

URL: http://www.springerlink.com/index/LHQHJGX3M7UNW4C2.pdf

Quellenhinweis: AS Neubauer, S Priglinger… - Der Ophthalmologe, 2001 - Springer

Titel: KAS, KIS, EKA, EPA, EGA, E-Health:--Ein Pladoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik

URL: http://www.imi.med.uni-erlangen.de/fileadmin/alt/team/download/mis_begriffsdefinitionen.pdf

Quellenhinweis: HU Prokosch - Informatik Biometrie und Epidemiologie in …, 2001 - imi.med.uni-erlangen.de

Titel: Die elektronische patientenakte–ist eine standardisierung in sicht

URL: http://cg.cs.tu-berlin.de/papers/2002vde_dgbmt.pdf

Quellenhinweis: S Märkle… - Tagungsband der Jahrestagung des VDE, 2002 - cg.cs.tu-berlin.de

Titel:

URL: http://scholar.google.de/scholar?cites=13968904875789183607&as_sdt=2005&sciodt=0,5&hl=en

Quellenhinweis: J Stadler - Ergebnisse einer Befragung von, 2002

Titel: Die Elektronische Patientenakte in der Intensivmedizin: Anforderungen-Konzepte-Nutzen

URL: http://www.telemedizinfuehrer.de/free/2004/stausberg_136_140.pdf

Quellenhinweis: J Stausberg, A Uslu… - Telemedizinführer …, 2004 - telemedizinfuehrer.de

Titel: Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker

URL: http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2006-926810

Quellenhinweis: C Jürgens, S Antal, F Heydenreich… - Klin Monatsbl …, 2006 - thieme-connect.com

Titel: Gesundheitstelematik-aktuelle Entwicklungen und Kosenquenzen für Krankenhäuser und Versorgungsverbände. Health Telematics-Current Developments and …

URL: http://www.thieme-connect.com/ejournals/abstract/klinikarzt/doi/10.1055/s-2003-44533

Quellenhinweis: SH Schug - Klinikarzt, 2003 - thieme-connect.com

Titel: Fortschritte in der geriatrischen Betreuung durch Telemedizin

URL: http://www.springerlink.com/index/95wrl605g114r001.pdf

Quellenhinweis: C Jürgens… - Der Ophthalmologe, 2006 - Springer

Titel: „Integration digitalisierter Filme und Bilder in die elektronische Patientenakte am Beispiel der Echokardiographie “

URL: http://www.kardio.org/?pub=37

| | |
|---|---|
| Quellenhinweis: | M Claus, HJ Appelrath, K Kronberg… - 2000 - kardio.org |
| Titel: | „Datenschutz und Telematik im Gesundheitswesen: Die Beispiele E-lektronisches Rezept und Elektronische Patientenakte" |
| URL: | http://www.ehealth.gvg-koeln.de/cms/medium/14/folien_mueller020330.pdf |
| Quellenhinweis: | J Müller - … und-gestaltung eV (GVG)(Hrsg.)(2002):„Health, 2002 - ehealth.gvg-koeln.de |
| Titel: | Eine objektorientierte Problembereichsanalyse für die elektronische Patientenakte |
| URL: | http://se.informatik.uni-oldenburg.de:30000/224/ |
| Quellenhinweis: | J Bodemann, W Hasselbring, D Mehlstäubler… - 1996 - se.informatik.uni-oldenburg.de |
| Titel: | Überwachung multipler Herzkreislaufparameter mittels Telemonitoring bei Patienten mit chronischer Herzinsuffizienz |
| URL: | http://www.springerlink.com/index/JH751K152W780404.pdf |
| Quellenhinweis: | M Oeff, P Kotsch, A Gößwald… - Herzschrittmachertherapie und …, 2005 - Springer |
| Titel: | Regionale elektronische Patientenakte am Beispiel einer ausgewählten Patientenklientel-Koronare Herzkrankheit |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-07-SchneiderP.pdf |
| Quellenhinweis: | P Schneider, G Knoll… - Proceedings des Nationalen …, 2007 - telemed-berlin.de |
| Titel: | Das konzept einer persönlichen elektronischen patientenakte im prepare-system |
| URL: | http://cg.cs.tu-berlin.de/papers/2001Telemed_PEMR.pdf |
| Quellenhinweis: | S Märkle - Tagungsband der Telemed, 2001 - cg.cs.tu-berlin.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=12295189903820134419&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Kolpatzik - 2005 - Verl. PCO |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=25491636231497925 83&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | K Schmidt - 2001):„Telemedizinführer Deutschland", Ausgabe, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=16603431424064936247&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | H Oswald, K Hafner… - Krankenhaus Umschau, 1996 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=14228007974081868974&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | HU Prokosch - Von Eiff W, Fenger H, Gillessen A, Kerres A, Mis U, … |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=16341527450930664499&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | AT im Gesundheitswesen |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=15897539282942036016&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Niemeyer, V Kanellopoulos-Niemeyer, F Paulsen… - Minerva Verlag |

| | |
|---|---|
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=15919543276925012785&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | G Brenner - Z Ärztl Fortbild Qual Gesundhwes, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=13360803221782854046&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | U Sax, A Weisbecker, J Falkner, F Viezens, HM YM… - E-HEALTH-COM 2007 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=2113228328585187733&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | A Mühlbacher… - 2003 - Technische Univ. |
| Titel: | Ablaufspezifikation durch Datenflußmodellierung und stromverarbeitende Funktionen |
| URL: | ftp://ftp.cs.tu-berlin.de/pub/local/korso/Nic93.ps.gz |
| Quellenhinweis: | F Nickl - 1993 - cs.tu-berlin.de |
| Titel: | IT-Unterstützung für den medizinischen Prozess in der integrierten Versorgung |
| URL: | http://www.gbv.de/dms/zbw/476629462.pdf |
| Quellenhinweis: | G Glock, S Sohn… - 2004 - gbv.de |
| Titel: | Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten am Beispiel des Tumorzentrums Heidelberg/Mannheim |
| URL: | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2006/6595/pdf/Zusammenfassung_Diss_Minne_van_der_Haak.pdf |
| Quellenhinweis: | M van_der_Haak - Medizinische Biometrie und …, 2006 - archiv.ub.uni-heidelberg.de |
| Titel: | Elektronische Erfassung von medizinischen Daten in deutschen Hausarztpraxen: Ein Telefon-Survey |
| URL: | http://www.allgemeinmedizin.med.uni-goettingen.de/literatur/heidenreich%20elektronische%20erfassung%20zaefq2005.pdf |
| Quellenhinweis: | R Heidenreich, W Himmel… - Z Ärztl …, 2005 - allgemeinmedizin.med.uni- … |
| Titel: | „meditrace: Zeitersparnis und Qualitätsverbesserung durch standardisierte, mobile Befunddokumentation" |
| URL: | http://www.medizin.uni-koeln.de/projekte/gmds-mocomed/workshop2001/tagungsband/10.pdf |
| Quellenhinweis: | M Walter, H Puhl… - 2001):„Mobiles Computing …, 2001 - medizin.uni-koeln.de |
| Titel: | Was versprechen Telemedizin und eHealth? Telematik-Bausteine für eine integrierte Versorgung-Entwicklungen auf deutscher und europäischer Ebene |
| URL: | http://www.kup.at/kup/pdf/3492.pdf |
| Quellenhinweis: | GTW Dietzel - Journal of Cardiology, 2003 - kup.at |
| Titel: | Portierung einer agentenbasierten elektronischen Patientenakte auf mobile Endgeräte |
| URL: | http://www.winfobase.de/lehrstuhl%5Cpublikat.nsf/intern01/E0420116A0B93A5EC125720D003CDB9D/$FILE/06-27.pdf |
| Quellenhinweis: | A Schweiger, C Hillebrand, A Sunyaev… - … der Mobiles Computing …, 2006 - winfobase.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=98439710961331214O&as_sdt=2005&sciodt=0,5&hl=en |

| | |
|---|---|
| Quellenhinweis: | F Warda - 2005 - Rheinware-Verl. |
| Titel: | Handbuch der medizinischen Informatik |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=pketYXl9lHkC&oi=fnd&pg=PA7&dq=%22elektronische+Patientenakte%22&ots=tkv-F_Unwt&sig=E642UbpLyDtC9QU45rBqDlYOZQk |
| Quellenhinweis: | TM Lehmann - 2004 - books.google.com |
| Titel: | Die unerträgliche Geschichte der GesundheitskarteinDeutschland |
| URL: | http://www.nachdenkseiten.de/upload/pdf/geschichte_der_e-card_Prof_G_Schweim_GMS.pdf |
| Quellenhinweis: | HG Schweim - 2007 - nachdenkseiten.de |
| Titel: | Informationsintegration in Gesundheitsversorgungsnetzen |
| URL: | http://www.springerlink.com/index/GL0502134X466236.pdf |
| Quellenhinweis: | R Lenz, M Beyer, C Meiler, S Jablonski… - Informatik-Spektrum, 2005 - Springer |
| Titel: | Mobile Klinische Arbeitsplätze Für Die Pflege: Aktuelle Möglichkeiten, Vorstellung Und Erprobung Einer Prototypentwicklung |
| URL: | http://www.medizin.uni-koeln.de/projekte/gmds-mocomed/workshop2001/tagungsband/16.pdf |
| Quellenhinweis: | M Urban… - Proceedings zum - medizin.uni-koeln.de |
| Titel: | Chancen für das Gesundheitswesen |
| URL: | http://www.aerzteblatt.de/pdf/99/21/a1417.pdf |
| Quellenhinweis: | GTW Dietzel - Dtsch Ärztebl, 2002 - aerzteblatt.de |
| Titel: | Die eGK-Lösungsarchitektur Architektur zur Unterstützung der Anwendungen der elektronischen Gesundheitskarte |
| URL: | http://www.springerlink.com/index/305k7233n85541x5.pdf |
| Quellenhinweis: | J Caumanns, H Weber, A Fellien, H Kurrek… - Informatik- …, 2006 - Springer |
| Titel: | „EPA. nrw “, ein Projekt der Landesregierung NRW mit Partnern aus Industrie und Selbst-verwaltung |
| URL: | http://www.telemedizinf%C3%BChrer.de/free/2008/kuehn_75_80.pdf |
| Quellenhinweis: | S Kühn - Jäckel, A.(Hg.): Telemedizinführer …, 2008 - xn--telemedizinfhrer-uzb.de |
| Titel: | Elektronische Patientenakte |
| URL: | http://epub.uni-regensburg.de/4664/ |
| Quellenhinweis: | B Blobel - 2005 - epub.uni-regensburg.de |
| Titel: | Partnership for the Heart. Klinische Erprobung eines telemedizinischen Betreuungssystems für Patienten mit chronischer Herzinsuffizienz |
| URL: | http://www.dgfkt.de/content/kardiotechnikoriginalausgaben/407/kt407luecke.pdf |
| Quellenhinweis: | F KÖHLER, S LÜCKE, A Kardiologie… - 2008 - dgfkt.de |
| Titel: | Aspekte der elektronischen Krankenakte in der Radiologie |
| URL: | http://www.springerlink.com/index/HHR6HU1BRQED1YQ7.pdf |
| Quellenhinweis: | K Adelhard, S Nissen-Meyer… - Der Radiologe, 1999 - Springer |
| Titel: | Politische Verantwortung bei der Entwicklung von Gesundheitstelematik und-informationssystemen |
| URL: | http://www.springerlink.com/index/MTMQ29D4LEHFYT2V.pdf |
| Quellenhinweis: | GTW Dietzel - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2003 - Springer |
| Titel: | Mehrwertdienste im Umfeld der elektronischen Gesundheitskarte |

| | |
|---|---|
| URL: | http://www.springerlink.com/index/16w052u268w56630.pdf |
| Quellenhinweis: | J Neuhaus, W Deiters… - Informatik-Spektrum, 2006 - Springer |
| Titel: | Die elektronische Patientenakte: ein internetbasiertes Konzept für das Management von Patientenbeziehungen |
| URL: | http://134.245.95.50:8080/dspace/handle/10419/36412 |
| Quellenhinweis: | A Mühlbacher… - 134.245.95.50 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=5174202365293567735&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | P Schmücker, C Ohr, A Beß, HB Bludau, R Haux… - … und Integration Informatik, …, 1998 |
| Titel: | Grid-basierte Services für die elektronische Patientenakte der Zukunft |
| URL: | http://en.scientificcommons.org/23781510 |
| Quellenhinweis: | U Sax, A Weisbecker, J Falkner, F Viezens… - 2007 - en.scientificcommons.org |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=11172558410503714 75&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | JH Müller - Dtsch Arztebl, 2008 |
| Titel: | Telemedizin aus medizinpsychologischer Perspektive–Der Einfluss von Telematikanwendungen auf die Arzt-Patientenbeziehung |
| URL: | http://iospress.metapress.com/index/7R3MD96NTC5FCAGC.pdf |
| Quellenhinweis: | S Schmidt, U Koch - Zeitschrift für Medizinische Psychologie, 2003 - IOS Press |
| Titel: | Bewertung des Patientenbuchs der Ärztlichen Qualitätsgemeinschaft Ried durch Hausärzte |
| URL: | http://www.adp-dormagen.dewww.pankreasinfo.com/fileadmin/medizinische_klinik/Abteilung_2/Sektion_Allgemeinmedizin/publikationen/Veroeffentlichungen/091_095.pdf |
| Quellenhinweis: | A Klingenberg, K Magdeburg… - Z Allg …, 2002 - adp-dormagen.dewww.pankreasinfo … |

## 6. Ergebnisse zum Suchbegriff "Gesundheitstelematik"

| | |
|---|---|
| Titel: | Telekommunikation, eCommunication |
| URL: | http://www.springerlink.com/index/p446p752523565p7.pdf |
| Quellenhinweis: | Haas … - Gesundheitstelematik, 2006 - Springer |
| Titel: | Gesundheits-Telematik—Rechtliche Antworten |
| URL: | http://www.springerlink.com/index/F30113346H0M13V7.pdf |
| Quellenhinweis: | C Dierks - Datenschutz und Datensicherheit-DuD, 2006 - Springer |
| Titel: | Politische Verantwortung bei der Entwicklung von Gesundheitstelematik und-informationssystemen |
| URL: | http://www.springerlink.com/index/MTMQ29D4LEHFYT2V.pdf |
| Quellenhinweis: | GTW Dietzel - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2003 - Springer |
| Titel: | |

| | |
|---|---|
| URL: | http://scholar.google.de/scholar?cites=10001088634136869638&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Boeker… - … : Vom Aufklärungsgespräch zur E-Health; Bern/ …, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=17033562012771684358&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | G Dietzel - … Deutschland. Ober-Mörlen: Medizin-Forum AG, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=2082063093999412780&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | N Frost - Eine interdisziplinäre Gegenstandsbeschreibung", …, 2000 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=6539766649731476450&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Brucksch - Aufl., Dietzenbach, S, 2003 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=12481584919351159999&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | UK Preusker - Deutsches Ärzteblatt |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=11771107753445036442&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Richer-Reichhelm - … und-gestaltung eV (GVG)(Hrsg.)(2002):„Health, 2002 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=15919543276925012785&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | G Brenner - Z Ärztl Fortbild Qual Gesundhwes, 2001 |
| Titel: | Kritische thesen zu patientenbezogenen anwendungen der gesundheitstelematik |
| URL: | http://www.springerlink.com/index/jp2r77162227v345.pdf |
| Quellenhinweis: | P Haas - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Gesundheitstelematik und Datenschutz |
| URL: | http://www.springerlink.com/index/l2740873x2136124.pdf |
| Quellenhinweis: | JH Müller - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Politische Aspekte und Ziele der Gesundheitstelematik |
| URL: | http://www.springerlink.com/index/x0r6v55642j12mh4.pdf |
| Quellenhinweis: | N Paland… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Akzeptanz der Gesundheitstelematik bei ihren Anwendern |
| URL: | http://www.springerlink.com/index/X1675520N8013710.pdf |
| Quellenhinweis: | S Schmidt, U Koch - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=18111823866552582 36&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | S Kirn… - Hohenheim, University of Hohenheim, 2005 |

| | |
|---|---|
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=8546288828376165149&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | HEG Krüger-Brand - Deutsches Ärzteblatt, 2006 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=18402535469256926922&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | U Hübner - Niederlag, Wolfgang/Rienhoff, Otto/Lemke, Heinz U., …, 2004 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=10097999049448203204&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | C Dierks, G Nitz… - MedizinRecht. de Verlag, Reihe Frankfurter Schriften, 2003 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=7492073882626475285&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | N Klusen… - 2002 - Nomos-Verl.-Ges |
| Titel: | Gesundheitstelematik: Datenmodelle und notwendige Infrastruktur |
| URL: | http://www.imi2.uni-luebeck.de/%7Eingenerf/publications/Inhalt%20FernUni%20Hagen%20Datenmodelle%20und%20notw%20Infrastrukturen.pdf |
| Quellenhinweis: | J Ingenerf… - Lehrbrief für das Nebenfach …, 2002 - imi2.uni-luebeck.de |
| Titel: | Telekommunikation, eCommunication |
| URL: | http://www.springerlink.com/index/p446p752523565p7.pdf |
| Quellenhinweis: | M wie Röntgenbilder… - Gesundheitstelematik, 2006 - Springer |
| Titel: | Gesundheits-Telematik—Rechtliche Antworten |
| URL: | http://www.springerlink.com/index/F30113346H0M13V7.pdf |
| Quellenhinweis: | C Dierks - Datenschutz und Datensicherheit-DuD, 2006 - Springer |
| Titel: | Politische Verantwortung bei der Entwicklung von Gesundheitstelematik und-informationssystemen |
| URL: | http://www.springerlink.com/index/MTMQ29D4LEHFYT2V.pdf |
| Quellenhinweis: | GTW Dietzel - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2003 - Springer |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=10001088634136869638&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Boeker… - … : Vom Aufklärungsgespräch zur E-Health; Bern/ …, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=17033562012771684358&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | G Dietzel - … Deutschland. Ober-Mörlen: Medizin-Forum AG, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=2082063093999412780&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | N Frost - Eine interdisziplinäre Gegenstandsbeschreibung", …, 2000 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=6539766649731476450&as_sdt=2005&sciodt=0, |

| | |
|---|---|
| | 5&hl=en |
| Quellenhinweis: | M Brucksch - Aufl., Dietzenbach, S, 2003 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=12481584919351159999&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | UK Preusker - Deutsches Ärzteblatt |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=11771107753445036442&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Richer-Reichhelm - … und-gestaltung eV (GVG)(Hrsg.)(2002):„Health, 2002 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=15919543276925012785&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | G Brenner - Z Ärztl Fortbild Qual Gesundhwes, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=4164089724941710375&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | H Krüger-Brand - Deutsches Ärzteblatt, 2004 |
| Titel: | Ökonomische Aspekte der Gesundheitstelematik |
| URL: | http://www.springerlink.com/index/t76178k27221k0u5.pdf |
| Quellenhinweis: | A Lux - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Gesundheitstelematik-Projekte in Deutschland aus Ländersicht |
| URL: | http://www.springerlink.com/index/l28227027674j134.pdf |
| Quellenhinweis: | SH Schug, M Redders - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Gesundheitstelematik/Telemedizin in der Republik Estland Telemedicine/e-Health in Estonia |
| URL: | http://www.thieme-connect.com/ejournals/abstract/dmw/doi/10.1055/s-2004-824838 |
| Quellenhinweis: | F Köhler, P Fotuhi, C Schierbaum… - Dtsch med …, 2004 - thieme-connect.com |
| Titel: | Diagnosen-und Prozedurendokumentation für Zwecke von DRGs, Qualitätsmanagement und Gesundheitstelematik |
| URL: | http://www.imi2.uni-luebeck.de/%7Eingenerf/publications/KIS2005%20Workshop1%20Hamburg.pdf |
| Quellenhinweis: | J Ingenerf, B Graubner… - KIS-Jahrestagung, …, 2005 - imi2.uni-luebeck.de |
| Titel: | Rahmenarchitektur und Sicherheitsinfrastruktur der deutschen Gesundheitstelematik-Plattform–die MDA Methodologie |
| URL: | http://www.telemedizinfuehrer.de/free/2005/blobel_89_96.pdf |
| Quellenhinweis: | B Blobel - Telemedizinführer Deutschland–Ausgabe, 2005 - telemedizinfuehrer.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=13997965731363798337&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | HE Kruger-Brand - … Mitteilungen-Ausgabe A, 2005 - Koln: Deutscher Arzte-Verlag GmbH, … |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=7827273463088074641&as_sdt=2005&sciodt=0, |

| | |
|---|---|
| | 5&hl=en |
| Quellenhinweis: | G Brenner - Zeitschrift fur Arztliche Fortbildung und …, 2001 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=4010967610812351392&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | CFJ Goetz - Zeitschrift fur Arztliche Fortbildung …, 2001 - Jena,[Germany]: Gustav Fischer, … |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=10053790986668474737&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | W Schlungbaum - Zeitschrift fur Arztliche …, 2001 - Jena,[Germany]: Gustav Fischer, … |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=17365010907556045946&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | N Klusen… - 2002 - Nomos |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=10345094765467684852&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | HE Krüger-Brand - Dtsch Arztebl, 2007 |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=16048358039771199877&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | C Brommelmeyer - Zeitschrift fur Arztliche …, 2001 - Jena,[Germany]: Gustav Fischer, … |
| Titel: | Rechtliche Aspekte der Gesundheitstelematik |
| URL: | http://www.springerlink.com/index/jw4g71w31453m81q.pdf |
| Quellenhinweis: | C Dierks - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Meilensteine auf dem Weg zur elektronischen Gesundheitskarte–Erfahrungen aus der Praxis |
| URL: | http://www.gmds.de/fachbereiche/informatik/WS_Gesundheitstelematik/WS07/07_Folien/Schablowski_Trautmann.pdf |
| Quellenhinweis: | M Schablowski-Trautmann, PT Infrastruktur… - … Gesundheitstelematik …, 2007 - gmds.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=661184500232075165&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | M Dietel… - Zeitschrift fur Arztliche Fortbildung und …, 2001 |
| Titel: | Lehren für eGK aus relevanten internationalen Strukturprojekten |
| URL: | http://ehealth.gvg-koeln.de/cms/medium/290/01_Blobel050503.pdf |
| Quellenhinweis: | B Blobel - 2005 - ehealth.gvg-koeln.de |
| Titel: | Sicheres und nachhaltiges eHealth |
| URL: | http://blog.eh-cc.de/wp-content/html/agdgi/down/KIS-2007_01_Blobel.pdf |
| Quellenhinweis: | B Blobel… - P. Schmücker, K.-H. Ellsässer (Hrsg.) - blog.eh-cc.de |
| Titel: | Die Implementierung von bIT4health im Spiegel der europäischen Initiativen sowie der |

| | |
|---|---|
| | fortgeschrittenen Programme anderer Länder |
| URL: | http://www.charite.de/medinfo/Userpages/Mitarbeiter/Stiller/CD/2005/Telemed-2005-09-Blobel-Vortrag.pdf |
| Quellenhinweis: | B Blobel - Steyer, Günter und Tolxdorff, Thomas,(eds.) bit for bit- …, 2005 - charite.de |
| Titel: | TELA–Telematik-Projektdatenbank für das deutsche Gesundheitswesen |
| URL: | http://www.telemedizinfuehrer.de/free/2004/schlutius_266_270.pdf |
| Quellenhinweis: | S Schlutius… - Telemedizinführer Deutschland - telemedizinfuehrer.de |

## 7. Ergebnisse zum Suchbegriff "Health Record"

| | |
|---|---|
| Titel: | … als Antwort auf Kommunikationsprobleme im Gesundheitswesen= A modular electronic health record as an answer to communication problems in health care |
| URL: | http://cat.inist.fr/?aModele=afficheN&cpsidt=16863321 |
| Quellenhinweis: | JC Schwarze, S Tessmann, C Sassenberg… - …, 2005 - cat.inist.fr |
| Titel: | A global socio-economic-medico-legal model for the sustainability of longitudinal electronic health records-part 2 |
| URL: | http://www.schattauer.de/en/magazine/subject-areas/journals-a-z/methods/contents/current-issue/issue/special/manuscript/5761.html |
| Quellenhinweis: | A Shabo - Methods of Information in Medicine, 2006 - schattauer.de |
| Titel: | KAS, KIS, EKA, EPA, EGA, E-Health:--Ein Pladoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik |
| URL: | http://www.imi.med.uni-erlangen.de/fileadmin/alt/team/download/mis_begriffsdefinitionen.pdf |
| Quellenhinweis: | HU Prokosch - Informatik Biometrie und Epidemiologie in …, 2001 - imi.med.uni-erlangen.de |
| Titel: | Assessment of middleware concepts using a generic component model |
| URL: | http://epub.uni-regensburg.de/4629/ |
| Quellenhinweis: | B Blobel - 1997 - epub.uni-regensburg.de |
| Titel: | Die elektronische patientenakte–ist eine standardisierung in sicht |
| URL: | http://cg.cs.tu-berlin.de/papers/2002vde_dgbmt.pdf |
| Quellenhinweis: | S Märkle… - Tagungsband der Jahrestagung des VDE, 2002 - cg.cs.tu-berlin.de |
| Titel: | e-Health |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=XgEJwTpaNn8C&oi=fnd&pg=PR15&dq=Health+Record&ots=_lkiP9oCQb&sig=xswDrvJbYUrfrNd6AKW5q55k1TE |
| Quellenhinweis: | K Jähn… - 2003 - books.google.com |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=5707395628930802678&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | T Kunze - … 2003. http://cybop. berlios. de/papers … |
| Titel: | Gender-specific factors in the utilization of medical services during adolescence* 1 |
| URL: | http://linkinghub.elsevier.com/retrieve/pii/S0140197196900688 |

| | |
|---|---|
| Quellenhinweis: | W Settertobulte… - Journal of adolescence, 1997 - Elsevier |
| Titel: | Neue Ansätze in der Erstversorgung von Herz-Kreislauf-Patienten |
| URL: | http://www.corscience.de/fileadmin/publikationen/Publikation_Bolz_neue_Ansaetze_Erstversorgung_2002.pdf |
| Quellenhinweis: | A Bob - corscience.de |
| Titel: | Environmental agents as cause of health disorders in children presented at an outpatient unit of environmental medicine |
| URL: | http://linkinghub.elsevier.com/retrieve/pii/S1438463904701613 |
| Quellenhinweis: | GA Wiesmüller, M Weishoff-Houben… - … environmental health, 2002 - Elsevier |
| Titel: | Digitale Krankenakte-Aspekte handlungsunterstützender klinischer Informationssysteme Digital Patient Records-Aspects of Action-supporting Clinical Information … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klinikarzt/doi/10.1055/s-2003-44530 |
| Quellenhinweis: | P Haas - Klinikarzt, 2003 - thieme-connect.com |
| Titel: | Die elektronische Gesundheitsakte in Deutschland |
| URL: | http://www.springerlink.com/index/NX737V0371444987.pdf |
| Quellenhinweis: | F Warda - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | Immune-stimulation improving health and performance; Immunstimulation zur Verbesserung der Gesundheit und Leistung |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2000/DE/DE00011.xml;DE1999A12947 |
| Quellenhinweis: | R Engstad… - Kraftfutter (Germany), 1999 - agris.fao.org |
| Titel: | An apple a day, the importance of apples in health promotion; An apple a day oder die gesundheitliche Bedeutung des Apfels |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2002/DE/DE02035.xml;DE2000H10383 |
| Quellenhinweis: | ME Hermann - Erwerbsobstbau (Germany), 2000 - agris.fao.org |
| Titel: | Sicheres und nachhaltiges eHealth |
| URL: | http://blog.eh-cc.de/wp-content/html/agdgi/down/KIS-2007_01_Blobel.pdf |
| Quellenhinweis: | B Blobel… - P. Schmücker, K.-H. Ellsässer (Hrsg.) - blog.eh-cc.de |
| Titel: | Die Implementierung von bIT4health im Spiegel der europäischen Initiativen sowie der fortgeschrittenen Programme anderer Länder |
| URL: | http://www.charite.de/medinfo/Userpages/Mitarbeiter/Stiller/CD/2005/Telemed-2005-09-Blobel-Vortrag.pdf |
| Quellenhinweis: | B Blobel - Steyer, Günter und Tolxdorff, Thomas,(eds.) bit for bit- …, 2005 - charite.de |
| Titel: | Aufbau einer einrichtungsübergreifenden Patientenakte in der Rhein-Neckar-Region |
| URL: | http://www.telemed-berlin.de/telemed2008/Medien/s152.pdf |
| Quellenhinweis: | O Heinze, A Brandner, R Brandner… - Telemed Tagungsband, …, 2008 - telemed-berlin.de |
| Titel: | Assessment of nutrition and health status of the elderly; Ermittlung des Ernaehrungs- und Gesundheitsstatus von Senioren |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2002/DE/DE02039.xml;DE2000O10471 |
| Quellenhinweis: | M Neuhäuser-Berthold, P Lührmann… - Aktuelle …, 2000 - agris.fao.org |

| | |
|---|---|
| Titel: | Wege zur elektronischen Patientenakte |
| URL: | http://www.springerlink.com/index/81L3541P42777QM6.pdf |
| Quellenhinweis: | B Blobel… - Datenschutz und Datensicherheit-DuD, 2006 - Springer |
| Titel: | EPA-Modelle im Vergleich: openEHR, HL7 V3 Specs, EN/ISO 13606, CCR |
| URL: | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-01-Blobel.pdf |
| Quellenhinweis: | B Blobel - Telemedizinführer Deutschland, 2008 - telemed-berlin.de |
| Titel: | Agentenbasierte elektronische Patientenakten |
| URL: | http://wwwkrcmar.in.tum.de/lehrstuhl%5Cpublikat.nsf/intern01/11EA8A14DB6EF7A0C12570D00027B24A/$FILE/05-30.pdf |
| Quellenhinweis: | A Schweiger, T Bastian… - Workshop der GMDS- … - wwwkrcmar.in.tum.de |
| Titel: | Softwareagenten für die Überwindung von Medienbrüchen bei der Patientenversorgung–ein Fallbeispiel aus dem Klinikum rechts der Isar der Technischen … |
| URL: | http://www.winfobase.de/lehrstuhl%5Cpublikat.nsf/intern01/7098CDA06E24F31CC12572120049FAC7/$FILE/06-35.pdf |
| Quellenhinweis: | A Schweiger, JM Leimeister… - HMD, Praxis der …, 2006 - winfobase.de |
| Titel: | openEHR–Die Geschichte eines Baukastensystems für eine gemeinsame Elektronische Gesundheitsakte |
| URL: | http://www.umi.cs.tu-bs.de/cms/staff/bergmann/bergmann_mdi_2005.pdf |
| Quellenhinweis: | J Bergmann… - umi.cs.tu-bs.de |
| Titel: | Herausforderungen bei der Umsetzung der elektronischen Patientenakte und Gesundheitskarte in Österreich |
| URL: | http://www.springerlink.com/index/Q542382884205150.pdf |
| Quellenhinweis: | KP Pfeiffer… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2009 - Springer |
| Titel: | Stand, Möglichkeiten und Grenzen der Telemedizin in Deutschland |
| URL: | http://www.springerlink.com/index/11121nk75w665044.pdf |
| Quellenhinweis: | R Klar… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2009 - Springer |
| Titel: | Kritische thesen zu patientenbezogenen anwendungen der gesundheitstelematik |
| URL: | http://www.springerlink.com/index/jp2r77162227v345.pdf |
| Quellenhinweis: | P Haas - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2005 - Springer |
| Titel: | The importance of nutrition in Public Health; Zur Bedeutung von Nutrition in der Public Health |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2002/DE/DE02012.xml;DE2000O10238 |
| Quellenhinweis: | R Manz… - Aktuelle Ernaehrungsmedizin (Germany), 2000 - agris.fao.org |
| Titel: | Indoor pyrethroid exposure in homes with woollen textile floor coverings |
| URL: | http://linkinghub.elsevier.com/retrieve/pii/S1438463904701789 |
| Quellenhinweis: | E Berger-Preifl, K Levsen, G Leng, H Idel… - … environmental health, 2002 - Elsevier |
| Titel: | Patientenzentrierte Dokumentation onkologischer Erkrankungen: Ein generisches XML-basiertes Informationsmodell zur syntaktischen und semantischen … |
| URL: | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2003/3818/pdf/Zusammenfassung_Astrid_Corinna_Wolff.pdf |
| Quellenhinweis: | A Wolff - Doktorarbeit, Ruprecht-Karls-Universität …, 2002 - archiv.ub.uni- |

| | |
|---|---|
| | heidelberg.de |
| Titel: | Lehren für eGK aus relevanten internationalen Strukturprojekten |
| URL: | http://ehealth.gvg-koeln.de/cms/medium/290/01_Blobel050503.pdf |
| Quellenhinweis: | B Blobel - 2005 - ehealth.gvg-koeln.de |
| Titel: | … zur Verbesserung der Versorgung oder Bedrohung hausärztlicher Arbeitsweise Disease Management Programm-Necessity to Improve Health Care or Threat to the … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/zfa/doi/10.1055/s-2006-942296 |
| Quellenhinweis: | N Schmacke - Z Allg Med, 2006 - thieme-connect.com |
| Titel: | Effect of microbial aerosols on human health; Wirkung von mikrobiellen Aerosolen auf den Menschen |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2000/DE/DE00024.xml;DE1999A12930 |
| Quellenhinweis: | C Herr, PM Bittighofer, J Bunger… - … , Reinhaltung der Luft ( …, 1999 - agris.fao.org |
| Titel: | Rahmenarchitektur und Sicherheitsinfrastruktur der deutschen Gesundheitstelematik-Plattform–die MDA Methodologie |
| URL: | http://www.telemedizinfuehrer.de/free/2005/blobel_89_96.pdf |
| Quellenhinweis: | B Blobel - Telemedizinführer Deutschland–Ausgabe, 2005 - telemedizinfuehrer.de |
| Titel: | Einsatz von CDA/SCIPHOX zur standardisierten Kommunikation medizinischer Befunddaten zwischen einem Schlaganfall-/Glaukom-Screening-Programm und … |
| URL: | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.3240&rep=rep1&type=pdf |
| Quellenhinweis: | F Gerdsen, S Müller, E Bader, M Poljak… - Proceedings der …, 2005 - Citeseer |
| Titel: | … operativer Eingriffe an einer Universitätsklinik: 3 Jahre klinische Erfahrung 1994-1996= The new German law of structural public health care (GSG) concerning … |
| URL: | http://cat.inist.fr/?aModele=afficheN&cpsidt=2120738 |
| Quellenhinweis: | FX Huber, R Schall, J Stern, U Gleim… - Das …, 1998 - cat.inist.fr |
| Titel: | Eating and nutrition within the concept of health-promoting schools; Essen und Ernaehrung im Konzept gesundheitsfoerdernder Schulen |
| URL: | http://agris.fao.org/agris-search/search/display.do?f=2002/DE/DE02005.xml;DE2000O10234 |
| Quellenhinweis: | I Heindl - Aktuelle Ernaehrungsmedizin (Germany), 2000 - agris.fao.org |
| Titel: | Sozio-kulturelle Aspekte in Health Technology Assessments (HTA) |
| URL: | http://linkinghub.elsevier.com/retrieve/pii/S1865921708000378 |
| Quellenhinweis: | A Gerhardus… - … für Evidenz, Fortbildung und Qualität im …, 2008 - Elsevier |
| Titel: | … von Menschen in der Aufenthaltsrechtlichen Illegalität: Deutschland und Italien im Vergleich. Health Problems and Health-Care among Unregistered Migrants: A … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/gesu/doi/10.1055/s-2007-1022527 |
| Quellenhinweis: | H Waller - Gesundheitswesen, 2008 - thieme-connect.com |
| Titel: | Was die philosophische Ontologie zur biomedizinischen Informatik beitragen kann |
| URL: | http://ontology.buffalo.edu/medo/WasDiePhilosophie.pdf |
| Quellenhinweis: | B Smith, D Siebert… - INFORMATION …, 2004 - ontology.buffalo.edu |
| Titel: | Informationsmanagement für vernetzte Versorgungsstrukturen |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=kYXnKXyzx94C&oi=fnd&pg= |

| | |
|---|---|
| | PA103&dq=Health+Record&ots=eCx9GbPOU0&sig=VMknMbv7v9wskeiRhxWDYHjc8JY |
| Quellenhinweis: | N Hellrung, R Haux… - Vernetzung im …, 2008 - books.google.com |
| Titel: | ByMedCard-An Electronic Patient Record with Chip Card Functionality |
| URL: | http://epub.uni-regensburg.de/4627/ |
| Quellenhinweis: | R Engelbrecht, V Böhm, C Hildebrand, W Moser… - 1997 - epub.uni-regensburg.de |
| Titel: | Strukturierte Abbildung radiologischer Leistungen im Kontext eines Krankenhausinformationssystems Bietet der DICOM-Standard hierfür die notwendigen … |
| URL: | http://www.springerlink.com/index/6N6M4KXXUDRLR6U9.pdf |
| Quellenhinweis: | H König… - Der Radiologe, 1998 - Springer |
| Titel: | Erweiterte Dienstleistungen einer Trusted Third Party für die elektronische Gesundheitsakte (Electronic Health Record) |
| URL: | http://epub.uni-regensburg.de/4692/ |
| Quellenhinweis: | B Blobel… - 2002 - epub.uni-regensburg.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar.bib?q=info:HlURY3tfExMJ:scholar.google.com/&output=citation&hl=en&as_sdt=0,5&ct=citation&cd=43 |
| Quellenhinweis: | DP Pretschner - Klin Monatsbl Augenheilkd, 2004 |
| Titel: | Literaturrecherche: Patient Health Record (PHR)-Die web-basierte Patienten-Akte |
| URL: | http://content.grin.com/document/v81357.pdf |
| Quellenhinweis: | CE Gerich - 2006 - GRIN Verlag |
| Titel: | Authorisation and Access Control in Distributed Electronic Health Record Systems |
| URL: | http://epub.uni-regensburg.de/4622/ |
| Quellenhinweis: | B Blobel… - 1999 - epub.uni-regensburg.de |
| Titel: | Die elektronische Gesundheitsakte (ELGA) in Österreich-Eine Evaluierung in Bezug auf funktionale Benutzeranforderungen. |
| URL: | http://www.nachwuchswissenschaftler.org/2010/1/28/ |
| Quellenhinweis: | A Ströher - German Journal for Young …, 2010 - nachwuchswissenschaftler.org |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:AzcWH3Nk2ZYJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | G Duftschmid, W Dorda, K Versorgung, W Gall… |
| Titel: | Secure HL7 for Communication in Distributed Health Information Systems |
| URL: | http://epub.uni-regensburg.de/4643/ |
| Quellenhinweis: | B Blobel, P Pharow… - 1998 - epub.uni-regensburg.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?q=related:AnUEK7k7X3gJ:scholar.google.com/&hl=en&as_sdt=0,5 |
| Quellenhinweis: | M Geisler - 2004 |

# 8. Ergebnisse zum Suchbegriff "Patient Record"

| | |
|---|---|
| Titel: | ByMedCard-An Electronic Patient Record with Chip Card Functionality |
| URL: | http://epub.uni-regensburg.de/4627/ |
| Quellenhinweis: | R Engelbrecht, V Böhm, C Hildebrand, W Moser… - 1997 - epub.uni-regensburg.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=4231299646471683569&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | RW Holl… - Mai: International Diabetes Monitor, 2000 |
| Titel: | Fluktuationen des intraokularen Augendrucks-ein intraindividueller Vergleich zwischen 24-Stunden-Teletonometrie-Monitoring und ambulanter Augendruckmessung … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2008-1027730 |
| Quellenhinweis: | C Jürgens, S Antal, K Henrici… - Klin Monatsbl …, 2009 - thieme-connect.com |
| Titel: | Prozesssimulation zur prospektiven Nutzwertanalyse einer voll digitalisierten Arbeitsumgebung am Beispiel eines sonographischen Arbeitsplatzes |
| URL: | http://www.thieme-connect.com/ejournals/abstract/roefo/doi/10.1055/s-2003-45332 |
| Quellenhinweis: | C Gillessen, UK Teichgräber, F Neumann… - Fortschr …, 2003 - thieme-connect.com |
| Titel: | KAS, KIS, EKA, EPA, EGA, E-Health:--Ein Pladoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik |
| URL: | http://www.imi.med.uni-erlangen.de/fileadmin/alt/team/download/mis_begriffsdefinitionen.pdf |
| Quellenhinweis: | HU Prokosch - Informatik Biometrie und Epidemiologie in …, 2001 - imi.med.uni-erlangen.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=13777416120746102078&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | V Schmidt, W Striebel, P Hufnagl, G Michelson… - Telemedizinführer …, 2001 |
| Titel: | Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2006-926810 |
| Quellenhinweis: | C Jürgens, S Antal, F Heydenreich… - Klin Monatsbl …, 2006 - thieme-connect.com |
| Titel: | Status and perspective of hospital information systems in Japan |
| URL: | http://www.schattauer.de/de/magazine/uebersicht/zeitschriften-a-z/methods/contents/current-issue/issue/714/manuscript/87.html |
| Quellenhinweis: | Y Haruki, Y Ogushi, Y Okada, M Kimura… - Methods of information …, 1999 - schattauer.de |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=13971916750393671858&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | J Stausberg… - Telemedizinführer Deutschland, 2006 |
| Titel: | Sprechen Sie LOINC |
| URL: | http://www.loinc.de/literatur/Einfuehrung_in_LOINC_%28HL7News8-2000%29.pdf |
| Quellenhinweis: | C McDonald, G Schadow, J Suico… - HL7-Mitteilungen, 2000 - loinc.de |
| Titel: | |

| | |
|---|---|
| URL: | http://scholar.google.de/scholar?cites=5323146806166569039&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | C Winkler - 2004 - Universitätsbibliothek |
| Titel: | Digitale Krankenakte-Aspekte handlungsunterstützender klinischer Informationssysteme Digital Patient Records-Aspects of Action-supporting Clinical Information … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klinikarzt/doi/10.1055/s-2003-44530 |
| Quellenhinweis: | P Haas - Klinikarzt, 2003 - thieme-connect.com |
| Titel: | The Treatment of Aspergillosis and Aspergilloma with Itraconazole, Clinical Results of an Open International Study (1982-1987)/Die Behandlung der Aspergillose und … |
| URL: | http://www3.interscience.wiley.com/journal/122343590/abstract |
| Quellenhinweis: | K Beule, P Doncker, G Cauwenbergh… - …, 1988 - interscience.wiley.com |
| Titel: | Aspekte der elektronischen Krankenakte in der Radiologie |
| URL: | http://www.springerlink.com/index/HHR6HU1BRQED1YQ7.pdf |
| Quellenhinweis: | K Adelhard, S Nissen-Meyer… - Der Radiologe, 1999 - Springer |
| Titel: | Sicheres und nachhaltiges eHealth |
| URL: | http://blog.eh-cc.de/wp-content/html/agdgi/down/KIS-2007_01_Blobel.pdf |
| Quellenhinweis: | B Blobel… - P. Schmücker, K.-H. Ellsässer (Hrsg.) - blog.eh-cc.de |
| Titel: | Elektronische oder papiergebundene Patientenakte Ein Kosten-Nutzen-Vergleich |
| URL: | http://www.springerlink.com/index/LHQHJGX3M7UNW4C2.pdf |
| Quellenhinweis: | AS Neubauer, S Priglinger… - Der Ophthalmologe, 2001 - Springer |
| Titel: | Integration mobiler Informationswerkzeuge in heterogene Krankenhausinformationssysteme. |
| URL: | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/1999/530/ |
| Quellenhinweis: | A Buchauer - 1999 - archiv.ub.uni-heidelberg.de |
| Titel: | Erfahrungsbericht über drei Jahre Routinebetrieb eines Anästhesie-Informations-Management-Systems (AIMS) am Universitätsklinikum Gießen |
| URL: | http://www.thieme-connect.com/ejournals/abstract/ains/doi/10.1055/s-1999-168 |
| Quellenhinweis: | M Benson, A Junger, L Quinzio… - Anästhesiol …, 1999 - thieme-connect.com |
| Titel: | Das konzept einer persönlichen elektronischen patientenakte im prepare-system |
| URL: | http://cg.cs.tu-berlin.de/papers/2001Telemed_PEMR.pdf |
| Quellenhinweis: | S Märkle - Tagungsband der Telemed, 2001 - cg.cs.tu-berlin.de |
| Titel: | Einheitlicher Zugriff auf klinische Daten in einem verteilten System Beispiel Radiologie |
| URL: | http://www.springerlink.com/index/L85NWV8AKRK9RB6H.pdf |
| Quellenhinweis: | K Adelhard, N Swoboda, S Nissen-Meyer… - Der Radiologe, 1999 - Springer |
| Titel: | Bonner Verlaufsbogen: Dokumentation der relevanten Parameter während der intravitrealen Medikamentenapplikation Bonner Record Form: Documentation of … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2008-1027475 |
| Quellenhinweis: | CH Meyer… - Klin Monatsbl Augenheilkd, 2008 - thieme-connect.com |
| Titel: | e-Health |
| URL: | http://books.google.de/books?hl=en&lr=lang_de&id=XgEJwTpaNn8C&oi=fnd&pg=PR15&dq=%22patient+record%22&ots=_lkiP9qCL9&sig=ehrO4zhXdxx_NSwRnKkHcgwzhhs |

| | |
|---|---|
| Quellenhinweis: | K Jähn… - 2003 - books.google.com |
| Titel: | Implementierung einer elektronischen Krankengeschichte in der Chirurgie |
| URL: | http://www.springerlink.com/index/C3C2UPGDCQNMY7WT.pdf |
| Quellenhinweis: | S Eggli… - Der Chirurg, 2001 - Springer |
| Titel: | Elektronische Erfassung von medizinischen Daten in deutschen Hausarztpraxen: Ein Telefon-Survey |
| URL: | http://www.allgemeinmedizin.med.uni-goettingen.de/literatur/heidenreich%20elektronische%20erfassung%20zaefq2005.pdf |
| Quellenhinweis: | R Heidenreich, W Himmel… - Z Ärztl …, 2005 - allgemeinmedizin.med.uni- … |
| Titel: | Die Kodierqualität in der stationären Versorgung |
| URL: | http://www.springerlink.com/index/FR535454155083J9.pdf |
| Quellenhinweis: | J Stausberg - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2007 - Springer |
| Titel: | Telemedizin: Chancen und Risiken |
| URL: | http://www.springerlink.com/index/BGMKV0PX9W2J9CC0.pdf |
| Quellenhinweis: | R Vogl - Der Radiologe, 2002 - Springer |
| Titel: | Einsatz einer elektronischen Patientenakte (EPA) an der Universitätsaugenklinik Magdeburg |
| URL: | http://www.springerlink.com/index/GYLCD4398H2DJTGR.pdf |
| Quellenhinweis: | J Kuchenbecker… - Der Ophthalmologe, 2004 - Springer |
| Titel: | Die Elektronische Assistentenhilfe: Arbeitsoptimierung und Fortbildungsinstrument |
| URL: | http://www.bullmed.ch/k2/pages/support/view.asp?k2dockey=C%3A%5CInetpub%5Cwwwroot%5Csaez%5Cpdf%5C2001%5C2001-12%5C2001-12-1201.PDF%40saez_d&serverSpec=schwabe03:9920&querytext=medizinische&OrigQuery=&QueryParser=Internet_Basic&logTitle=Die+elektronische+Assistentenhilfe%3A+%24%24+Arbeitsoptimierung+und+Fortbildungsinstrument&dtype=3 |
| Quellenhinweis: | S Eggli, P Imesch, D Schölly… - Schweiz Ärztezeitung, 2001 - bullmed.ch |
| Titel: | Was die philosophische Ontologie zur biomedizinischen Informatik beitragen kann |
| URL: | http://ontology.buffalo.edu/medo/WasDiePhilosophie.pdf |
| Quellenhinweis: | B Smith, D Siebert… - INFORMATION …, 2004 - ontology.buffalo.edu |
| Titel: | Die klinische Archivierung-Anforderungen und aktuelle Lösungen |
| URL: | http://www.thieme-connect.com/ejournals/abstract/zblgyn/doi/10.1055/s-2000-10099 |
| Quellenhinweis: | P Schmücker, O Reinhard… - Zentralbl Gynakol, 2000 - thieme-connect.com |
| Titel: | Implementierung einer elektronischen Krankenakte |
| URL: | http://www.springerlink.com/index/V883TH4253748J5P.pdf |
| Quellenhinweis: | A Grüner, A Ljutow, W Schleinzer… - Der Schmerz, 2008 - Springer |
| Titel: | Strukturierte Abbildung radiologischer Leistungen im Kontext eines Krankenhausinformationssystems Bietet der DICOM-Standard hierfür die notwendigen … |
| URL: | http://www.springerlink.com/index/6N6M4KXXUDRLR6U9.pdf |
| Quellenhinweis: | H König… - Der Radiologe, 1998 - Springer |
| Titel: | Die Operation nach Keller-Brandes: Langzeitergebnisse bei jungen Patienten mit Hallux valgus |
| URL: | http://www.fussforum.at/html/pdf/Brandes1999.pdf |
| Quellenhinweis: | A Zembsch, HJ Trnka, G Menschik… - Z Orthop, 1999 - fussforum.at |

| | |
|---|---|
| Titel: | Stete Weiterentwicklung nötig-Integration heterogener IT-Systeme im Krankenhaus Constant Further Development is Necessary-Integration of Heterogeneous IT … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klinikarzt/doi/10.1055/s-2003-44532 |
| Quellenhinweis: | R Lenz… - Klinikarzt, 2003 - thieme-connect.com |
| Titel: | Zur Frage der Handlungsfahigkeit bei penetrierender und stumpfer Bauchverletzung |
| URL: | http://www.springerlink.com/index/q1306553078v6051.pdf |
| Quellenhinweis: | KS Saternus, G Bessel… - International Journal of Legal Medicine, 1983 - Springer |
| Titel: | Tagesschwankungen des okulären Perfusionsdrucks im Telemonitoring bei primärem Offenwinkelglaukom Diurnal Variation of Ocular Pressure in Open-Angle … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2008-1027918 |
| Quellenhinweis: | S Antal, C Jürgens, R Großjohann… - Klin Monatsbl …, 2009 - thieme-connect.com |
| Titel: | Agentenbasierte elektronische Patientenakten |
| URL: | http://wwwkrcmar.in.tum.de/lehrstuhl%5Cpublikat.nsf/intern01/11EA8A14DB6EF7A0C12570D00027B24A/$FILE/05-30.pdf |
| Quellenhinweis: | A Schweiger, T Bastian… - Workshop der GMDS- … - wwwkrcmar.in.tum.de |
| Titel: | Abteilungs-und Patientenmanagement in der Strahlenheilkunde |
| URL: | http://www.springerlink.com/index/V3812281838RP453.pdf |
| Quellenhinweis: | F Heinemann, F Röhner, M Schmucker… - Strahlentherapie und …, 2009 - Springer |
| Titel: | Häufigkeit der nosokomialen Pneumonie im Krankenhaus-Vergleich von rechnergestützter Basisdokumentation und papierbasierter Krankenakte Frequency of … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/pneumologie/doi/10.1055/s-2008-1038099 |
| Quellenhinweis: | A Azaouagh… - Pneumologie, 2008 - thieme-connect.com |
| Titel: | Aufbau einer einrichtungsübergreifenden Patientenakte in der Rhein-Neckar-Region |
| URL: | http://www.telemed-berlin.de/telemed2008/Medien/s152.pdf |
| Quellenhinweis: | O Heinze, A Brandner, R Brandner… - Telemed Tagungsband, …, 2008 - telemed-berlin.de |
| Titel: | Detaillierte Erfassung von Inanspruchnahme, Morbidität, Erkrankungsverläufen und Ergebnissen durch episodenbezogene Dokumentation in der Hausarztpraxis … |
| URL: | http://www.thieme-connect.com/ejournals/abstract/gesu/doi/10.1055/s-2007-976517 |
| Quellenhinweis: | G Laux, T Rosemann, T Körner… - …, 2007 - thieme-connect.com |
| Titel: | Elektronische Befund-und Bildverteilung aus einem PACS: Umsetzung der datenschutzrechtlichen A spekte am Beispiel des Universitätsklinikums Freiburg |
| URL: | http://www.thieme-connect.com/ejournals/abstract/roefo/doi/10.1055/s-2003-39919 |
| Quellenhinweis: | E Kotter, D Jäger, M Binder, A Roesner… - Fortschr …, 2003 - thieme-connect.com |
| Titel: | Proaktive Assistenz zur kontextabhängigen und zielorientierten Unterstützung bei der Indikationsstellung und Anwendung von Behandlungsmaßnahmen in der … |
| URL: | http://134.130.184.8/opus/volltexte/2009/2688/ |
| Quellenhinweis: | M Sedlmayr - 2008 - 134.130.184.8 |
| Titel: | Qualitätssicherung mit teleradiologie |
| URL: | http://www.springerlink.com/index/WGDC5QD7CTL05YXW.pdf |
| Quellenhinweis: | M Walz, G Weisser, R Bolte, J Teubner, R Loose… - Der Radiologe, 2002 - Springer |
| Titel: | Ein graph-und objektorientiertes Patientendaten-Modell |

| | |
|---|---|
| URL: | http://www.staff.uni-mainz.de/pommeren/Artikel/patmod.pdf |
| Quellenhinweis: | R Müller… - 1994 - staff.uni-mainz.de |
| Titel: | PAGE-Eine Plattform zur Integration IT-basierter Gesundheitsdienstleistungen in Gesundheitsnetzwerke |
| URL: | http://www.vde-verlag.de/proceedings-en/453138074.html |
| Quellenhinweis: | N Hellrung, M Gövercin, R Haux, A Hein… - … Assisted Living-AAL, 2009 - vde-verlag.de |
| Titel: | Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten |
| URL: | http://www.ssk.med.uni-erlangen.de/images/Presentations/vanderhaak-skonetzki_2006_berlin_telemed_abstract.pdf |
| Quellenhinweis: | M van der Haak, P Knaup-Gregori… - Proceedings zur …, 2006 - ssk.med.uni-erlangen.de |
| Titel: | Kommunikation und Integration medizinischer Dokumente für Telekonferenz-Anwendungen |
| URL: | http://www.thieme-connect.com/ejournals/abstract/roefo/doi/10.1055/s-2008-1032540 |
| Quellenhinweis: | R Felix, L Kleinholz, H Oswald, M Ohly… - Fortschr …, 1994 - thieme-connect.com |
| Titel: | Diagnosehäufigkeiten und Verordnungen bei Schwindel im Patientenkollektiv einer hausärztlichen Routinedatenbank |
| URL: | http://linkinghub.elsevier.com/retrieve/pii/S1865921708001293 |
| Quellenhinweis: | C Kruschinski, M Kersting, A Breull… - Zeitschrift für Evidenz, …, 2008 - Elsevier |
| Titel: | |
| URL: | http://scholar.google.de/scholar?cites=17422998432730992188&as_sdt=2005&sciodt=0,5&hl=en |
| Quellenhinweis: | P Haas, H Markus, OJ Bott, J Bergmann, M Walter… - gms german medical …, 2008 |

## 9. Einschränkung der Suchergebnisse

Nach der ersten Sichtung der Suchergebnisse aus Anhang B, Punkte 1 bis 8 sind die folgenden Treffer zu Systemen in der engeren Auswahl verblieben. Treffer, denen keine Systembezeichnung zugeordnet ist, sind nachträglich aus der Untersuchung ausgeschlossen, weil es sich entweder nicht um die Beschreibung eines vollständigen Systems, sondern nur eines bestimmten Aspekts handelt oder weil das beschriebene System nicht als verteilte Krankenakte klassifizierbar ist.

**Systembezeichnung: EPA. Nrw**

| | |
|---|---|
| Suchbegriff | „care record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] „EPA. nrw “, ein Projekt der Landesregierung NRW mit Partnern aus Industrie und Selbstverwaltung<br>[PDF] from xn--telemedizinfhrer-uzb.de; S Kühn - Jäckel, A.(Hg.): Telemedizinführer …, 2008 - xn--telemedizinfhrer-uzb.de<br>... Datenmodellen • HL7 RIM – Reference Information Model • CEN ENV 13606 – Electronic Health-<br>care Record Communication • openEHR • IHE XDS Profile – Integrating the Healthcare Enterprise •<br>HL7 V2 MDM -Nachrichten • ANSI CCR – Continuity of Care Record … |
| URLs | http://www.telemedizinführer.de/free/2008/kuehn_75_80.pdf |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „care record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Systematische Analyse und Bewertung transmuraler Schnittstellen der Tiroler e-Health-Aktivitäten anhand des IHE IT-I-Frameworks<br>[PDF] from umit.atME des Akademischen Grades… - iig.umit.at<br>... Grundlagen 20 summaries like ASTM's Continuity of Care Record (CCR) or HL7's Continuity<br>of Care Document (CCD), is owned by the patient and has patient input and access that spans<br>episodes of care across multiple CDOs within a community, region, or state (or … |
| URLs | http://iig.umit.at/dokumente/msc_stark.pdf |

**Systembezeichnung: Münster:Diss:Konzepte zur elektronischen Arztbriefschreibung und–Übermittlung**

| | |
|---|---|
| Suchbegriff | „care record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Konzepte zur elektronischen Arztbriefschreibung und–Übermittlung<br>[PDF] from uni-muenster.deK im Gesundheitswesen - miami.uni-muenster.de<br>Page 1. Aus dem Universitätsklinikum Münster Institut für Medizinische Informatik und Biomathematik – Direktor: Prof. Dr. W. Köpcke – Konzepte zur elektronischen Arztbriefschreibung und –Übermittlung - Verbesserung der interinstitutionellen ... |
| URLs | http://miami.uni-muenster.de/servlets/DerivateServlet/Derivate-4823/diss_butta.pdf |

**Systembezeichnung: Braunschweig:Bergmann: An eConsent-based system architecture supporting cooperation in integrated healthcare networks**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | An eConsent-based system architecture supporting cooperation in integrated healthcare networks<br>[PDF] from unisalento.it J Bergmann, OJ Bott, I Hoffmann… - Studies in health …, 2005 - IOS Press<br>... An eConsent-based approach assures, that access to the distributed health record remains under control of the patient. ... The patient authorizes a health professional, who is involved into the care process, to access the shared distributed health record. … |
| URLs | http://iospress.metapress.com/index/7e64lx91hl1e9c5v.pdf und<br>Joachim Bergmann et al., “An e-consent-based shared EHR system architecture for integrated healthcare networks,” *International Journal of Medical Informatics* 76, no. 2-3 (February): 130-136, doi:10.1016/j.ijmedinf.2006.07.013. |

**Systembezeichnung: Konzept: Content-based medical image retrieval in peer-to-peer systems**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Content-based medical image retrieval in peer-to-peer systems<br>A Charisi… - … of the 1st ACM International Health …, 2010 - portal.acm.org<br>Page 1. Content-Based Medical Image Retrieval in Peer-to-Peer Systems Amalia Charisi<br>Computer Engineering and Informatics Department University of Patras 26500 Patras, Greece charisa@ceid.upatras.gr Vasileios Megalooikonomou … |
| URLs | http://portal.acm.org/citation.cfm?id=1883103 |

**Systembezeichnung: Beispielsystem: Prag: EuroMISE Center: Universal Electronic Health Record MUDR ("Multimedia Distributed Record")**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Universal Electronic Health Record MUDR<br>M Duplaga - IOS Press<br>... health documentation of a patient. 1. Architecture of MUDR EHR Following these requirements, a modular structure of a system called MUDR (Multimedia distributed health record) has been defined. The main architecture of the ... |
| URLs | http://iospress.metapress.com/index/J8Y28CURGNNQW9G3.pdf |

**Systembezeichnung: Beispielsystem: Ontology-based distributed health record management system**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Ontology-based distributed health record management system<br>[PDF] from utcluj.roC Cenan, G Sebestyen, G Saplacan… - … and Processing, 2008 …, 2008 - ieeexplore.ieee.org<br>The paper presents the architecture of a distributed software system that manages patients' health records and assures remote and interactive access to medical services. |

| | |
|---|---|
| | In order to cover the complex relationships between different medical concepts (symptoms, diseases, ... |
| URLs | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4648393 |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Security requirements and solutions in distributed electronic health records<br>B Blobel - Computers and Security, 1997 - ingentaconnect.com<br>... system. Distributed health record systems must meet high demands for data protection and data security, which concern integrity, availability, onfidentiality including access management, and accountability. Communication … |
| URLs | http://www.ingentaconnect.com/content/els/01674048/1997/00000016/00000003/art84526 |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „distributed health record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 1 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Consumer-centric and privacy-preserving identity management for distributed e-health systems<br>R Au… - … on System Sciences, Proceedings of the 41st …, 2008 - ieeexplore.ieee.org<br>... user privacy. Sensitive medical information can be collected from distributed health record databases in different clinic/hospitals and linked together dynamically without revealing the consumer's real identity. The architectural … |
| URLs | http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4438938<br>oder http://dx.doi.org/10.1109/HICSS.2008.101 |

**Systembezeichnung: elektronische Fallakte**

| | |
|---|---|
| Suchbegriff | „elektronische Fallakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Die elektronische Fallakte-ein Standard für die einrichtungsübergreifende Kommunikation<br>C Reuter, J Neuhaus, J Caumanns… - 2009 - en.scientificcommons.org<br>... 45722961. Die elektronische Fallakte - ein Standard für die einrichtungsübergreifende Kommunikation (2009). ... Repository, Fraunhofer Publica (Germany). Keywords, elektronische Fallakte, einrichtungsübergreifende Kommunikation, Gesundheitswesen, Standard. … |
| URLs | http://en.scientificcommons.org/45722961<br>http://www.fallakte.de, http://www.fallakte.de/ueber-efa,<br>http://www.fallakte.de/faq#welcheinformationenenthaelt |

**Systembezeichnung: elektronische Fallakte**

| | |
|---|---|
| Suchbegriff | „elektronische Fallakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [CITATION] eFA-Die Elektronische Fallakte als Basis für sektorübergreifende Prozesse<br>J Neuhaus… - 2007 - en.scientificcommons.org<br>Publication View. 28076252. eFA - Die Elektronische Fallakte als Basis für sektorübergreifende Prozesse (2007). … |

| | |
|---|---|
| URLs | http://en.scientificcommons.org/28076252<br>www.fallakte.de bzw. http://www.fallakte.de/ueber-efa,<br>http://www.fallakte.de/faq#welcheinformationenenthaelt |

**Systembezeichnung: elektronische Fallakte**

| | |
|---|---|
| Suchbegriff | „elektronische Fallakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Föderatives Identitätsmanagement am Beispiel der elektronischen Fallakte<br>O Boehm… - Informatik-Spektrum, 2007 - Springer<br>... Die elektronische Fallakte ist somit eine vom Patienten bewilligte, integrierte Sicht auf die Daten der verschiedenen Leistungserbringer, die für die gemeinsame Behandlung relevant sind. Architektur In den Anwendungsszenarien … |
| URLs | http://www.springerlink.com/index/L4518084790K1V73.pdf<br>www.fallakte.de bzw. http://www.fallakte.de/ueber-efa,<br>http://www.fallakte.de/faq#welcheinformationenenthaelt |

**Systembezeichnung: M.Eckenbach:NRW:Dortmund:D2D:Mamma@kte.nrw**

| | |
|---|---|
| Suchbegriff | „elektronische Fallakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | D2D-MammaAkte der KV Nordrhein<br>G Mohr, E Gehlen… - 54. Kongress der …, 2003 - books.google.com<br>... Implementationen in überschaubarer Zahl von verschiedenen Praxissystemen Stufe 2 „Indikationsbezogene Fallakte (eMammaAkte)" Darüber hinaus soll im Herbst 2002 der Startschuss fallen für eine indikationsbezogene elektronische Fallakte (Indikation „ ... |
| URLs | http://books.google.de/books?hl=en&lr=lang_de&id=qMv8BxX7MewC&oi=fnd&pg=PA263&dq=%22elektronische+Fallakte%22&ots=yppTe8xPoI&sig=1v5Vvg7B6matpGV9aaK_TvhHnac<br>http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.3988&rep=rep1&type=pdf#page=42 |

**Systembezeichnung: elektronische Fallakte**

| | |
|---|---|
| Suchbegriff | „elektronische Fallakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Elektronische Fallakten zur sicheren einrichtungsübergreifenden Kommunikation [PDF] from e-fallakte.deJ Caumanns, O Boehm… - 2007 - e-fallakte.de<br>... In Abbildung 1 ist die elektronische Fallakte als arztgeführte, einrichtungsübergreifend nutzbare Akte anderen Formen der elektronischen Verwaltung von Patientendaten gegenübergestellt. ... Elektronische Fallakte … Medizinische Dokumentation … |
| URLs | http://www.e-fallakte.de/Elektronische_FallAkten_zur_sicheren_einrichtungs%C3%BCbergreifenden_Kooperation.pdf,<br>http://www.fallakte.de, http://www.fallakte.de/ueber-efa,<br>http://www.fallakte.de/faq#welcheinformationenenthaelt |

**Systembezeichnung: M.Eckenbach:NRW:Dortmund:D2D:Mamma@kte.nrw**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |

| | |
|---|---|
| Kommentar | [PDF] Integration von Anwendungssystemen des stationären und ambulanten Versorgungssektors am Beispiel des Projektes Mamma@ kte. nrw<br>[PDF] from psu.eduP Haas, M Eckenbach, H Plagge… - EAI 2005 Enterprise … - Citeseer<br>... Dabei sollten folgende Randbedin- gungen berücksichtigt werden: • Basis für die Elektronische<br>Krankenakte sollte die sichere Server-Lösung der KV Nordrhein D2D (Doctor to Doctor) bzw.<br>die technische Lösung PaDok der Fraun- hofer Gesellschaft St. … |
| URLs | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.3988&rep=rep1&type=pdf#page=42 |

**Systembezeichnung: Greifswald:Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker Teletonometrie in Mecklenburg-Vorpommern**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker<br>C Jürgens, S Antal, F Heydenreich… - Klin Monatsbl …, 2006 - thieme-connect.com<br>... werden. Die hier vorgestellte EPA kann also in bestehende Lösungen integriert werden. Seitenanfang. Ergebnisse. Für jeden der zwischenzeitlich erfassten 120 Patienten wird eine elektronische Krankenakte angelegt. Die bei ... |
| URLs | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-2006-926810 |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Implementierung einer elektronischen Krankenakte<br>A Grüner, A Ljutow, W Schleinzer… - Der Schmerz, 2008 - Springer<br>... Schmerzklinik. Hier bei wurde auf der Grundlage eines kommerziellen Systems eine eigene, bedarfsadaptierte, elektronische Krankenakte geschaffen und die Papierakte in einem längeren Prozess durch das EDVSystem er setzt. … |
| URLs | http://www.springerlink.com/index/V883TH4253748J5P.pdf |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Die Implementierung von bIT4health im Spiegel der europäischen Initiativen sowie der fortgeschrittenen Programme anderer Länder<br>[PDF] from charite.de B Blobel - Steyer, Günter und Tolxdorff, Thomas,(eds.) bit for bit- …, 2005 - charite.de<br>... über ein Nationales Institut für Gesundheitstelematik, welches die eHealth Prozesse leitet und koordiniert. Alle Länder außer Deutschland priorisieren die elektronische Krankenakte, wobei die einzelnen Länder durchaus unterschiedliche Ausprägungen verfolgen. Page 24. … |
| URLs | http://www.charite.de/medinfo/Userpages/Mitarbeiter/Stiller/CD/2005/Telemed-2005-09-Blobel-Vortrag.pdf |

**Systembezeichnung: Oldenburg:Konzept:Verteilte Architekturen zur intra- und inter-institutionellen Integration von Patientendaten**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Verteilte Architekturen zur intra-und inter-institutionellen Integration von Patientendaten<br>[PDF] from uni-oldenburg.deL Bischofs, W Hasselbring, H Niemann… - 2004 - se.informatik.uni-oldenburg.de<br>... Methoden H.-U. Prokosch [4] definiert die Begriffe elektronische Krankenakte (EKA) und elektronische Patientenakte (EPA) wie folgt: Die EKA beinhaltet die strukturierte, medizinische Dokumentation innerhalb eines Krankenhausinformationssystems. … |
| URLs | http://se.informatik.uni-oldenburg.de:30000/73/1/gmds2004.pdf |

**Systembezeichnung: Erlangen:Augenklinik:Webbasierte elektronische Krankenakte im Rahmen eines integrierten Versorgungskonzepts als Instrument der Qualitätssicherung**

| | |
|---|---|
| Suchbegriff | „elektronische Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Webbasierte elektronische Krankenakte im Rahmen eines integrierten Versorgungskonzepts als Instrument der Qualitätssicherung Web-Based Electronic Patient …<br>A Händel, AGM Jünemann… - Klin Monatsbl …, 2009 - thieme-connect.com<br>Ziel: Wesentliche Voraussetzung für die Integrierte Versorgung (IGV) ist der Aufbau eines Kommunikationsnetzwerks als Basis für die sektorenübergreifende Qualitätssicherung. Vor diesem Hintergrund eines IGV-Vertrags der Universitäts-Augenklinik Erlangen mit der AOK Bayern war es das Ziel, … |
| URLs | http://www.thieme-connect.com/ejournals/abstract/klimo/doi/10.1055/s-0028-1109193 (Quelle nicht Verfügbar) |

**Systembezeichnung: Suhl:Regionale elektronische Patientenakte am Beispiel einer ausgewählten Patientenklientel-Koronare Herzkrankheit**

| | |
|---|---|
| Suchbegriff | „elektronische Patientenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Regionale elektronische Patientenakte am Beispiel einer ausgewählten Patientenklientel-Koronare Herzkrankheit<br>P Schneider, G Knoll… - Proceedings des Nationalen …, 2007 - telemed-berlin.de<br>Hier wird die informationstechnologische Infrastruktur der SRH Zentralklinikum Suhl gGmbH unter Einbeziehung der Kooperationspartner (Soll / Ist – Darstellung) aufgezeigt. Gleichzeitig erfolgen Aussagen zur Prozessoptimierung und zur Workflow-Darstellung. |
| URLs | http://www.telemed-berlin.de/telemed2007/Beitraege/TELEMED-2007-07-SchneiderP.pdf |

**Systembezeichnung: Heidelberg:TheorKonzept: Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten**

| | |
|---|---|
| Suchbegriff | „elektronische Patientenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |

| | |
|---|---|
| Kommentar | Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten am Beispiel des Tumorzentrums Heidelberg/Mannheim<br>M van_der_Haak - Medizinische Biometrie und …, 2006 - archiv.ub.uni-heidelberg.de<br>... Eine einrichtungsübergreifende elektronische Patientenakte, welche alle relevanten medizinischen Daten eines gemeinsam behandelten Patienten enthält, soll die Kommunikation zwischen den beteiligten Einrichtungen unterstützen und eine optimale Patientenversorgung ... |
| URLs | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2006/6595/pdf/Zusammenfassung_Diss_Minne_van_der_Haak.pdf<br>http://www.ssk.med.uni-erlangen.de/images/Presentations/vanderhaak-skonetzki_2006_berlin_telemed_abstract.pdf |

**Systembezeichnung: Sax:TheorKonzept:Grid-basierte Services für die elektronische Patientenakte der Zukunft**

| | |
|---|---|
| Suchbegriff | „elektronische Patientenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [CITATION] Grid-basierte Services für die elektronische Patientenakte der Zukunft<br>U Sax, A Weisbecker, J Falkner, F Viezens… - 2007 - en.scientificcommons.org<br>Publication View. 23781510. Grid-basierte Services für die elektronische Patientenakte der Zukunft (2007). … |
| URLs | http://en.scientificcommons.org/23781510 |

**Systembezeichnung: Beispiel zum Einsatz von CDA/SCIPHOX zur standardisierten Kommunikation medizinischer Befunddaten zwischen einem Schlaganfall-/Glaukom-Screening-Programm und einer elektronischen Gesundheitsakte**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Einsatz von CDA/SCIPHOX zur standardisierten Kommunikation medizinischer Befunddaten zwischen einem Schlaganfall-/Glaukom-Screening-Programm und …<br>[PDF] from psu.eduF Gerdsen, S Müller, E Bader, M Poljak… - Proceedings der …, 2005 - Citeseer<br>... Patient Empowerment and the Electronic Health Record. ... 6. Ückert F, Müller ML, Bürkle T, Prokosch HU: An electronic health record to support patients and institutions of the health care system<br>German medical science 2 (2004): 1-11 7. Waegemann, CP (1999). … |
| URLs | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.3240&rep=rep1&type=pdf<br>http://www.telemed-berlin.de/telemed2005/pdf/S%20Gerdsen.pdf |

**Systembezeichnung: AkteOnline des Universitätsklinikums Münster**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | … als Antwort auf Kommunikationsprobleme im Gesundheitswesen= A modular electronic health record as an answer to communication problems in health care<br>JC Schwarze, S Tessmann, C Sassenberg… - …, 2005 - cat.inist.fr<br>Since 2000 the University of Muenster has developed an electronic health record (EHR) called akteonline.de. Several clinics and departments use the EHR in routine. akteonline.de in its current structure supports patients as well as health care professionals and aims at providing a … |
| URLs | http://cat.inist.fr/?aModele=afficheN&cpsidt=16863321 |

| | |
|---|---|
| | http://campus.uni-muenster.de/3775.html |

**Systembezeichnung: Architektur für "Agentenbasierte elektronische Patientenakten" (Lehrstuhl Krcmar, TUM)**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Softwareagenten für die Überwindung von Medienbrüchen bei der Patientenversorgung–ein Fallbeispiel aus dem Klinikum rechts der Isar der Technischen … [PDF] from winfobase.deA Schweiger, JM Leimeister… - HMD, Praxis der …, 2006 - winfobase.de ... 5. Electronic Health Record: Der Zugang zu den medizinischen Daten wird durch den mündigen Bürger kontrolliert, mit der Möglichkeit, selbstständig Daten zu seiner Gesundheitsakte hinzuzufügen. Das Klinikum rechts der Isar befindet sich in dieser Ordnung auf Stufe 2. … |
| URLs | http://www.winfobase.de/lehrstuhl%5Cpublikat.nsf/intern01/7098CDA06E24F31CC12572120049FAC7/$FILE/06-35.pdf http://wwwkrcmar.in.tum.de/lehrstuhl%5Cpublikat.nsf/intern01/11EA8A14DB6EF7A0C12570D00027B24A/$FILE/05-30.pdf |

**Systembezeichnung: Beispiel zum Einsatz von CDA/SCIPHOX zur standardisierten Kommunikation medizinischer Befunddaten zwischen einem Schlaganfall-/Glaukom-Screening-Programm und einer elektronischen Gesundheitsakte**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Einsatz von CDA/SCIPHOX zur standardisierten Kommunikation medizinischer Befunddaten zwischen einem Schlaganfall-/Glaukom-Screening-Programm und … [PDF] from psu.edu F Gerdsen, S Müller, E Bader, M Poljak… - Proceedings der …, 2005 – Citeseer ... Patient Empowerment and the Electronic Health Record. ... 6. Ückert F, Müller ML, Bürkle T, Prokosch HU: An electronic health record to support patients and institutions of the health care system German medical science 2 (2004): 1-11 7. Waegemann, CP (1999). … |
| URLs | http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.3240&rep=rep1&type=pdf http://www.telemed-berlin.de/telemed2005/pdf/S%20Gerdsen.pdf |

**Systembezeichnung: Patientenakte in der Rhein-Neckar-Region (Projekt ISIS (Intersektorales Informationssystem)) UK Heidelberg**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Aufbau einer einrichtungsübergreifenden Patientenakte in der Rhein-Neckar-Region [PDF] from telemed-berlin.deO Heinze, A Brandner, R Brandner… - Telemed Tagungsband, …, 2008 - telemed-berlin.de ... Diese neue Generation von Akten heißt elektronische Gesundheitsakte (EGA) oder personal health record (PHR) und lässt zusätzlich zur eEPA die Eingabe von Inhalten zu bspw. Wellness, Ernährung Page 2. Schug S, Engelmann U (Hrsg). Telemed 2008 Proceedings. … |
| URLs | http://www.telemed-berlin.de/telemed2008/Medien/s152.pdf |

**Systembezeichnung: openEHR**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] openEHR–Die Geschichte eines Baukastensystems für eine gemeinsame Elektronische Gesundheitsakte<br>[PDF] from tu-bs.deJ Bergmann… - umi.cs.tu-bs.de<br>... In Anlehnung an den Begriff des Shared Care kann diese Form der Dokumentation als Shared EHR (Shared Electronic Health Record) bezeichnet werden. Bei der Patientenversorgung werden zwar auch zukünftig die eigenen Aufzeichnungen einer Einrichtung die ... |
| URLs | http://www.umi.cs.tu-bs.de/cms/staff/bergmann/bergmann_mdi_2005.pdf<br>http://www.openehr.org/home.html |

**Systembezeichnung: Patientenzentrierte Dokumentation onkologischer Erkrankungen: Ein generisches XML-basiertes Informationsmodell … (Heidelberg)**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Patientenzentrierte Dokumentation onkologischer Erkrankungen: Ein generisches XML-basiertes Informationsmodell zur syntaktischen und semantischen …<br>A Wolff - Doktorarbeit, Ruprecht-Karls-Universität …, 2002 - archiv.ub.uni-heidelberg.de<br>... Anhand dieser konnten die vier in- ternationalen Standards für elektronische Patientenakten Electronic Health Care Record Architecture (EHCRA), Clinical Document Architecture (CDA), Good Electronic Health Record (GEHR) und Medical Markup Language (MML) bewertet ... |
| URLs | http://archiv.ub.uni-heidelberg.de/volltextserver/volltexte/2003/3818/pdf/Zusammenfassung_Astrid_Corinna_Wolff.pdf |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Rahmenarchitektur und Sicherheitsinfrastruktur der deutschen Gesundheitstelematik-Plattform–die MDA Methodologie<br>[PDF] from telemedizinfuehrer.de B Blobel - Telemedizinführer Deutschland–Ausgabe, 2005 - telemedizinfuehrer.de<br>... Sie dient als Versichertenkarte, als Impfausweis, als elektronisches Rezept, zur Speicherung von Verweisen auf Kom- ponenten der elektronischen Patientenakte (Electronic Patient Record – EPR oder Electronic Health Record – EHR) bzw. … |
| URLs | http://www.telemedizinfuehrer.de/free/2005/blobel_89_96.pdf |

**Systembezeichnung: Architektur für "Agentenbasierte elektronische Patientenakten" (Lehrstuhl Krcmar, TUM)**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Agentenbasierte elektronische Patientenakten<br>[PDF] from tum.de A Schweiger, T Bastian… - Workshop der GMDS- … - wwwkrcmar.in.tum.de<br>... Electronic Health Record: Der Zugang zu den medizinischen Daten wird durch den |

| | |
|---|---|
| | mündigen Bürger kontrolliert. ... Empowerment of patients and communication with health care professionals through an electronic health record. … |
| URLs | http://wwwkrcmar.in.tum.de/lehrstuhl%5Cpublikat.nsf/intern01/11EA8A14DB6EF7A0C12570D00027B24A/$FILE/05-30.pdf |

**Systembezeichnung: Österreich:Projekt:ELGA**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Herausforderungen bei der Umsetzung der elektronischen Patientenakte und Gesundheitskarte in Österreich<br>KP Pfeiffer… - Bundesgesundheitsblatt-Gesundheitsforschung- …, 2009 - Springer<br>... Es soll jedoch auch möglich sein, dass der Bürger seinen persönlichen Ge- sundheitsakt (Personal Health Record) im Rahmen von ELGA führt. ... ÖKZ 48:6–11 4. Feasibility study for implementing the electronic health record (ELGA) in the Austrian health system. … |
| URLs | http://www.springerlink.com/index/Q542382884205150.pdf<br>http://www.initiative-elga.at/ |

**Systembezeichnung: ByMedCard und ByMedConnect (experimentell, Helmholz)**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | ByMedCard-An Electronic Patient Record with Chip Card Functionality<br>R Engelbrecht, V Böhm, C Hildebrand, W Moser… - 1997 - epub.uni-regensburg.de<br>... und Böhm, V. und Hildebrand, C. und Moser, W. und Landgraf, R. und Hierl, F. und Töppel, S.<br>und Blobel, Bernd und Diedrich, T. (1997) ByMedCard - An Electronic Patient Record with Chip Card Functionality. In: van den Broek, Laurens und Sikkel, AJ, (eds.) Health cards '97. … |
| URLs | http://epub.uni-regensburg.de/4627/<br>http://epub.uni-regensburg.de/4627/; http://www.helmholtz-muenchen.de/medizinische-informationssysteme-medis/projekte-der-arbeitsgruppe-medis/unterstuetzungssysteme/bymedcard/index.html |

**Systembezeichnung: keine, nicht ausreichend Konkret um vergleichbar zu sein**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Rahmenarchitektur und Sicherheitsinfrastruktur der deutschen Gesundheitstelematik-Plattform–die MDA Methodologie<br>[PDF] from telemedizinfuehrer.de B Blobel - Telemedizinführer Deutschland–Ausgabe, 2005 - telemedizinfuehrer.de<br>... Sie dient als Versichertenkarte, als Impfausweis, als elektronisches Rezept, zur Speicherung von Verweisen auf Kom- ponenten der elektronischen Patientenakte (Electronic Patient Record – EPR oder Electronic Health Record – EHR) bzw. … |
| URLs | http://www.telemedizinfuehrer.de/free/2005/blobel_89_96.pdf |

**Systembezeichnung: Blobel: Beispiel für Sichere Kommunikation mit HL7**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 1 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [CITATION] Secure HL7 for Communication in Distributed Health Information |

| | |
|---|---|
| | Systems<br>B Blobel, P Pharow… - 1998 - epub.uni-regensburg.de<br>... In: Waegemann, CP, (ed.) Toward An Electronic Health Record Europe ´98. Medical Record Institute, Newton, MA., S. 210-214. ... Zusätzliche Informationen (Öffentlich): Towards an Electronic Health Record Europe, 15-18 November, 1998, London, |
| URLs | http://epub.uni-regensburg.de/4643/ (Literatur nicht verfügbar) |

**Systembezeichnung: keine, beschreibt kein konkretes System oder Systemmodell**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 0 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | A global socio-economic-medico-legal model for the sustainability of longitudinal electronic health records-part 2<br>A Shabo - Methods of Information in Medicine, 2006 - schattauer.de<br>... Results: The vision is that lifetime EHRs should be sustained by new players in the healthcare arena, who will function as independent health record banks (IHRBs). Multiple competing IHRBs would be established and regulated following preemptive legislation. ... |
| URLs | http://www.schattauer.de/en/magazine/subject-areas/journals-a-z/methods/contents/current-issue/issue/special/manuscript/5761.html |

**Systembezeichnung: keine, beschreibt kein konkretes bewertbares System**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 0 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [CITATION] Assessment of middleware concepts using a generic component model<br>B Blobel - 1997 - epub.uni-regensburg.de<br>... In: Waegemann, C. Peter, (ed.) Toward an Electronic Health Record Europe '97: Conference on the Creation of a European Electronic Health Record: 19 - 22 October, Cumberland Hotel London. Medical Records Institute, Newton, Mass., S. 221-228. ISBN 0-9640667-3-4. … |
| URLs | |

**Systembezeichnung: keine, beschreibt kein konkretes bewertbares System**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 0 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | ] KAS, KIS, EKA, EPA, EGA, E-Health:--Ein Pladoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik<br>[PDF] from uni-erlangen.deHU Prokosch - Informatik Biometrie und Epidemiologie in …, 2001 - imi.med.uni-erlangen.de<br>... In diesem Kontext ist auch der von Waegemann [12] beschriebene electronic health record zu sehen, den man ins Deutsche als elektronische Gesundheitsakte übersetzen kann. ... In: Toward<br>an Electronic Health Record ´99, Medical Records Institute, 1999, 116-118. … |
| URLs | |

**Systembezeichnung: keine, beschreibt kein konkretes bewertbares System**

| | |
|---|---|
| Suchbegriff | „Health Record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 0 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Die elektronische patientenakte–ist eine standardisierung in sicht<br>[PDF] from tu-berlin.deS Märkle… - Tagungsband der Jahrestagung des VDE, 2002 - |

| | |
|---|---|
| | cg.cs.tu-berlin.de<br>... wurden aus einem großen Fun- dus von Projektinformationen relevante publizierte Ergebnisse von zahlreichen, hauptsächlich europäischen Forschungs- und Entwicklungsprojekten ausgewählt und untersucht: - GEHR – The Good European Health Record Project [15 … |
| URLs | |

**Systembezeichnung: Magdeburg:Augenklinik:nur Abteilungssystem:Einsatz einer elektronischen Patientenakte (EPA) an der Universitätsaugenklinik Magdeburg**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Einsatz einer elektronischen Patientenakte (EPA) an der Universitätsaugenklinik Magdeburg<br>J Kuchenbecker… - Der Ophthalmologe, 2004 - Springer<br>... Schlüsselwörter Elektronische Patientenakte · EPA · KKS · PACS · DICOM Abstract Background.The electronic or paper- based patient record is the most important documentation file. The electronic patient record has many advantages compared to a paper-based record. … |
| URLs | http://www.springerlink.com/index/GYLCD4398H2DJTGR.pdf |

**Systembezeichnung: München:Großhadern:InterneVerteilung:Einheitlicher Zugriff auf klinische Daten in einem verteilten System Beispiel Radiologie**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Einheitlicher Zugriff auf klinische Daten in einem verteilten System Beispiel Radiologie<br>K Adelhard, N Swoboda, S Nissen-Meyer… - Der Radiologe, 1999 - Springer<br>... Literatur 1. Ball MJ, Collen MF (1992) (eds) Aspects of the computer-based patient record. ...<br>Int J Biomed Comput 29:169–189 7. Dick RS, Steen EB (eds) (1997) The Computer-based Patient Record.<br>An Essential Technology for Health Care, 2nd edn. ... |
| URLs | http://www.springerlink.com/index/L85NWV8AKRK9RB6H.pdf |

**Systembezeichnung: keine, außerhalb der Definition verteilter Krankenakten**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Elektronische Befund-und Bildverteilung aus einem PACS: Umsetzung der datenschutzrechtlichen Aspekte am Beispiel des Universitätsklinikums Freiburg<br>E Kotter, D Jäger, M Binder, A Roesner… - Fortschr …, 2003 - thieme-connect.com<br>... The system was installed in January 2001 and the experience of its two years in operation is reported. Seitenanfang. Key words. PACS - report and image distribution - data privacy - electronic patient record. Seitenanfang. |
| URLs | http://www.thieme-connect.com/ejournals/abstract/roefo/doi/10.1055/s-2003-39919 |

**Systembezeichnung: Berlin:BERMED (Paper von 1994)**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Kommunikation und Integration medizinischer Dokumente für Telekonferenz- |

| | |
|---|---|
| | Anwendungen<br>R Felix, L Kleinholz, H Oswald, M Ohly… - Fortschr …, 1994 - thieme-connect.com<br>... the realisation of computer-based medical conferencing. This requires the integration of the multimedia data in form of a meta-patient record for our medical application systems. These applications are supported by a distributed … |
| URLs | http://www.thieme-connect.com/ejournals/abstract/roefo/doi/10.1055/s-2008-1032540 |

**Systembezeichnung: Heidelberg:TheorKonzept: Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten<br>[PDF] from uni-erlangen.deM van der Haak, P Knaup-Gregori… - Proceedings zur …, 2006 - ssk.med.uni-erlangen.de<br>... 4. JH van Bemmel, Toward a virtual electronic patient record, MD Comput 1999,16,20-21. ...<br>6. DW Forslund, RL Phillips, DG Kilman, JL Cook, Experiences with a distributed virtual patient record system, Proc AMIA Annu Fall Symp 1996,86,483-487. … |
| URLs | http://www.ssk.med.uni-erlangen.de/images/Presentations/vanderhaak-skonetzki_2006_berlin_telemed_abstract.pdf |

**Systembezeichnung:**

| | |
|---|---|
| Suchbegriff | „patient record“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 2 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | Implementierung einer elektronischen Krankenakte<br>A Grüner, A Ljutow, W Schleinzer… - Der Schmerz, 2008 - Springer<br>... Schlüsselwörter Elektronische Krankenakte · Interdisziplinäre Dokumentation · Qualitätssicherung · Schmerztherapie Implementation of an electronic patient record. ... A complex electronic patient record was im- plemented in an interdisciplinary pain clinic. . |
| URLs | http://www.springerlink.com/index/V883TH4253748J5P.pdf |

**Systembezeichnung: Regensburg:Klinikum:Einweiserportal**

| | |
|---|---|
| Suchbegriff | „verteilte Krankenakte“ |
| Suchmaschine | Google Scholar (de) |
| Relevanz | 3 (Schätzung bei 1. Durchsicht. In Punkten 1-3) |
| Kommentar | [PDF] Generischer Architekturansatz für Telemedizin Portale und verteilte Krankenakten<br>[PDF] from uni-regensburg.de W Wiedermann, A Tsakpinis… - 2008 - epub.uni-regensburg.de<br>... erweiterbaren Systemstruktur anbieten. Mikrokernel: Grundprinzip der vorliegenden Softwarearchitektur ist eine, dem Anwendungsfall „Verteilte Krankenakte“ angepasste Mikrokernel-Architektur. Der Nutzen einer Mikrokernel ... |
| URLs | http://epub.uni-regensburg.de/8909/1/s065.pdf |

# 10. Ergebnisse der Bewertung der einzelnen Systeme

**ChiperMe: Personal Electronic Health Records in Patients hands**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: keine Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: keine Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Unbekannt |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Funktionalität |
| Menge der Knoten mit eigener Datenspeicherung | Zentrale Datenspeicherung: ein zentraler Knoten der die Daten speichert |
| Redundanz | Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbasiert |
| Verbreitungsgebiet der Krankenakte | Testbetrieb |

**AkteOnline des Universitätsklinikums Münster**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: teilweise Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: teilweise Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Unbekannt |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Zentrale Datenspeicherung: ein zentraler Knoten der die Daten speichert |
| Redundanz | Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbasiert |
| Verbreitungsgebiet der Krankenakte | Verbund: rechtlich abgegrenzter Verbund von Einrichtungen |

**Architektur für "Agentenbasierte elektronische Patientenakten" (Lehrstuhl Krcmar, TUM)**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |

| | |
|---|---|
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Software-Agenten |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Peer to Peer |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Nicht im Einsatz (Konzept) |

**Beispielsystem: Ontology-based distributed health record management system**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: vollständige Integration |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Schichtenarchitektur |
| Inhaltlicher Umfang der Akte | Funktionsbezogene partielle Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Unbekannt |

**Beispielsystem: Prag: EuroMISE Center: Universal Electronic Health Record MUDR ("Multimedia Distributed Record")**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Programmcode |
| Art der Erweiterbarkeit und Veränderbarkeit | Proprietäre Konfiguration, veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: proprietär |
| Dominierende Muster der Softwarearchitektur | Schichtenarchitektur |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Funktionalität |
| Menge der Knoten mit eigener Datenspeicherung | Unbekannt |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Nicht im Einsatz (Konzept) |

**Berlin:BERMED (Paper von 1994)**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: vollständige Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Keine Redundanz: Dokument liegt ausschließlich in einem Systemknoten vor |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Nicht im Einsatz (Konzept) |

**Braunschweig:Bergmann: An eConsent-based system architecture supporting cooperation in integrated healthcare networks**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Vollständige Prozessintegration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Komponentenbasierte Architektur |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Keine Redundanz: Dokument liegt ausschließlich in einem Systemknoten vor |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbasiert |
| Verbreitungsgebiet der Krankenakte | Regionaler Verbund: Regionaler abgegrenzter Verbund von Einrichtungen |

**ByMedCard und ByMedConnect (experimentell, Helmholz)**

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: teilweise Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: teilweise Integration |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Message Broker |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |

| | |
|---|---|
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Vollständige Redundanz: Dokument liegt in jedem Systemknoten vor |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Regionaler Verbund: Regionaler abgegrenzter Verbund von Einrichtungen |

**elektronische Fallakte**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: teilweise Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: teilweise Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Serviceorientierte Architektur |
| Inhaltlicher Umfang der Akte | Fallbezogene partielle Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Keine Redundanz: Dokument liegt ausschließlich in einem Systemknoten vor |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Fallbasiert |
| Verbreitungsgebiet der Krankenakte | Verbund: rechtlich abgegrenzter Verbund von Einrichtungen |

**EPA. Nrw**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Vollständige Prozessintegration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Unbekannt |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Regional: Alle Einrichtungen einer Region |

### GEHR - The Good European Health Record Project

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Programmcode |
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: vollständige Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: teilweise Integration |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Mikrokernel-Architektur |
| Inhaltlicher Umfang der Akte | Institutionsbezogene partielle Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Funktionalität |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbestandteilbasiert |
| Verbreitungsgebiet der Krankenakte | Testbetrieb |

### Greifswald:Elektronische Patientenakte zum telemedizinischen Monitoring von Augeninnendruck, Blutdruck und Blutzucker Teletonometrie in Mecklenburg-Vorpommern

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Funktionsbezogene partielle Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Zentrale Datenspeicherung: ein zentraler Knoten der die Daten speichert |
| Redundanz | Keine Redundanz: Dokument liegt ausschließlich in einem Systemknoten vor |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Patientenbasiert |
| Verbreitungsgebiet der Krankenakte | Regionaler Verbund: Regionaler abgegrenzter Verbund von Einrichtungen |

### Heidelberg:TheorKonzept: Architekturkonzepte für einrichtungsübergreifende elektronische Patientenakten

| *Bewertungskriterium* | *Ausprägung* |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: keine Integration |

| | |
|---|---|
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: keine Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Fallbezogene partielle Krankenakte |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Verbund: rechtlich abgegrenzter Verbund von Einrichtungen |

**Konzept: Content-based medical image retrieval in peer-to-peer systems**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Bilddokument: Bildsequenz: Mit Metadaten |
| Dominierende Muster der Softwarearchitektur | Sonstige |
| Inhaltlicher Umfang der Akte | Unbekannt |
| Kommunikationsform | Pull, Synchron |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Peer to Peer |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Nicht im Einsatz (Konzept) |

**M.Eckenbach:NRW:Dortmund:D2D:Mamma@kte.nrw**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: teilweise Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: teilweise Integration |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Message Broker |
| Inhaltlicher Umfang der Akte | Funktionsbezogene partielle Krankenakte |
| Kommunikationsform | Push, Asynchron, ein Sender, mehrere klar adressierte Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Vollständige Redundanz: Dokument liegt in jedem Systemknoten vor |
| Rollen der Kommunikanten | Client / Server |

| | |
|---|---|
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Regionaler Verbund: Regionaler abgegrenzter Verbund von Einrichtungen |

**openEHR**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Vollständige Prozessintegration |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Serviceorientierte Architektur |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Pull, Synchron |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbestandteilbasiert |
| Verbreitungsgebiet der Krankenakte | International (einzeln): Verschiedene Einrichtungen weltweit |

**Österreich:Projekt:ELGA**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Umfassende Krankenakte |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Vollständig dezentrale Datenspeicherung: jeder Knoten speichert seine Daten selbst |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | National: Alle Einrichtungen eines Landes |

**Patientenakte in der Rhein-Neckar-Region (Projekt ISIS (Intersektorales Informationssystem)) UK Heidelberg**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Anbieterseitig: vollständige Integration |
| Art der Integration in bestehende System- und Prozesslandschaft | Nutzerseitig: keine Integration |
| Art der verfügbaren Dokumente und Inhalte | Bilddokument: Bildsequenz: Mit Metadaten |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |

| | |
|---|---|
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Unbekannt |
| Kommunikationsform | Push, Asynchron, ein Sender, ein Empfänger |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Funktionalität |
| Menge der Knoten mit eigener Datenspeicherung | Zentrale Datenspeicherung: ein zentraler Knoten der die Daten speichert |
| Redundanz | Redundanz zur Veröffentlichung: Primärsystem und zentrale Speicherung |
| Rollen der Kommunikanten | Client / Server |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Dokumentenbasiert |
| Verbreitungsgebiet der Krankenakte | Regionaler Verbund: Regionaler abgegrenzter Verbund von Einrichtungen |

**Patientenzentrierte Dokumentation onkologischer Erkrankungen: Ein generisches XML-basiertes Informationsmodell . (Heidelberg)**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | standardisierte Konfiguration veränderbar und erweiterbar |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Strukturiertes Dokument: standardisiert |
| Dominierende Muster der Softwarearchitektur | Unbekannt |
| Inhaltlicher Umfang der Akte | Funktionsbezogene partielle Krankenakte |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Datenstrukturen |
| Menge der Knoten mit eigener Datenspeicherung | Unbekannt |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Unbekannt |

**Sax:TheorKonzept:Grid-basierte Services für die elektronische Patientenakte der Zukunft**

| ***Bewertungskriterium*** | ***Ausprägung*** |
|---|---|
| Art der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Art der Integration in bestehende System- und Prozesslandschaft | Unbekannt |
| Art der verfügbaren Dokumente und Inhalte | Beliebige Dokumente |
| Dominierende Muster der Softwarearchitektur | Komponentenbasierte Architektur |
| Dominierende Muster der Softwarearchitektur | Serviceorientierte Architektur |
| Inhaltlicher Umfang der Akte | Unbekannt |
| Kommunikationsform | Unbekannt |
| Kontext der Erweiterbarkeit und Veränderbarkeit | Unbekannt |
| Menge der Knoten mit eigener Datenspeicherung | Dezentrale Server: mehrere spezialisierte Knoten speichern die Daten |
| Redundanz | Unbekannt |
| Rollen der Kommunikanten | Unbekannt |
| Tiefe bzw. Detaillierungsgrad des Autorisierungskonzepts | Unbekannt |
| Verbreitungsgebiet der Krankenakte | Nicht im Einsatz (Konzept) |

# Anhang C: Auschnitt aus HL7 CDA

## HL7 CDA Element StructuredBody

```
<xs:complexType name="POCD_MT000020.StructuredBody">
      <xs:sequence>
            <xs:element name="confidentialityCode" type="CE" minOccurs="0"/>
            <xs:element name="languageCode" type="CS" minOccurs="0"/>
            <xs:element name="component" type="POCD_MT000020.Component6"
                        maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="type" type="Classes" default="ActHeir"/>
      <xs:attribute name="classCode" type="ActClass" use="optional"
                    default="DOCBODY"/>
      <xs:attribute name="moodCode" type="ActMood" use="optional"
                     default="EVN"/>
      <xs:attribute name="templateId" use="optional">
            <xs:simpleType>
                  <xs:list itemType="oid"/>
            </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="typeID" use="optional">
            <xs:simpleType>
                  <xs:list itemType="oid"/>
            </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="realmCode" use="optional">
            <xs:simpleType>
                  <xs:list itemType="cs"/>
            </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="nullFlavor" type="cs" use="optional"/>
</xs:complexType> (Dolin u. a. 2004:0)
```

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Literaturverzeichnis

Abadi, Martín & Cardelli, Luca 1996. *A theory of objects*. Springer.

Abran, Alain u. a. 2004. *Guide to the Software Engineering Body of Knowledge (SWEBOK)*. IEEE Press. http://www.computer.org/portal/web/swebok/htmlformat [Stand 2011-04-29].

Alexander, Christopher 1979. *The timeless way of building*. Oxford University Press US.

Alexander, Christopher, Ishikawa, Sara & Silverstein, Murray 1977. *A pattern language*. Oxford University Press US.

Allen, Christopher & Dierks, Tim 1999. RFC 2246 - The TLS Protocol Version 1.0, Internet Engineering Task Force (IETF). http://tools.ietf.org/html/rfc2246 [Stand 2011-01-15].

Anderson, Ross J. 2010. *Security Engineering*. Second Edition. Indianapolis: John Wiley & Sons.

Atkinson, R. & Kent, S. 1998. RFC 2401 - Security Architecture for the Internet Protocol, Internet Engineering Task Force (IETF). http://tools.ietf.org/html/rfc2401 [Stand 2011-01-15].

Bass, Len, Clements, Paul & Kazman, Rick 2003. *Software architecture in practice*. Addison-Wesley.

Bayer, Alexander u. a. 2007. Web Plattform der Universitätsklinik Erlangen für die Umsetzung telemedizinischer Anforderungen und Dokumentation Integrierter Versorgungssysteme. In G. Steyer & T. Tolxdorff *TELEMED 2007 – Electronic Health Record und Gesundheitsportale*. Telemed 2007. Berlin.

Beale, Thomas u. a. 2006. *Introducing openEHR*. http://www.openehr.org/releases/1.0.2/openEHR/introducing_openEHR.pdf [Stand 2011-02-11].

Beale, Thomas u. a. 2007. *The openEHR Reference Model - EHR Information Model, Revision 5.1.0*. http://www.openehr.org/releases/1.0.1/architecture/rm/ehr_im.pdf [Stand 2011-02-11].

Beale, Thomas & Heard, Sam 2007. Archetype Definition Language ADL 2. http://www.openehr.org/releases/1.0.2/architecture/am/adl2.pdf [Stand 2010-12-17].

Beale, Thomas & Heard, Sam 2008. openEHR Architecture, Architecture Overview, Revision 1.1.1. http://www.openehr.org/releases/1.0.2/architecture/overview.pdf [Stand 2013-02-11].

Beck, Kent & Cunningham, Ward 1987. Using Pattern Languages for Object-Oriented Programs. In *Technical Report No. CR-87-43*. OOPSLA ´87 workshop on Specification and Design for Object-Oriented Programming. http://c2.com/doc/oopsla87.html [Stand 2012-05-4].

Beck, Klaus 2007. *Kommunikationswissenschaft*. 2., überarb. Auflage. Stuttgart: UTB.

Bergler, Tobias 2008. Case Management - ein Erfahrungsbericht aus ärztlicher Sicht. In 1. Regensburger Case Management Symposium 2008. Regensburg. http://www.uniklinikum-regensburg.de/imperia/md/content/ueber-uns/zentraleabteilungen/pflegedirektion/cm_kongress_2008_tobias_bergler.pdf [Stand 2013-01-27].

Bergmann, Joachim u. a. 2007. An e-consent-based shared EHR system architecture for integrated healthcare networks. *International Journal of Medical Informatics* 76, 2-3, 130–136. http://www.sciencedirect.com/science/article/B6T7S-4KW60SV-1/2/eb60a9b6afcad03ab34b064c2bf0a6aa [Stand 2011-03-3].

Bergmann, Joachim u. a. 2005. Klinische Befundportale als Weg zu einer Konsens-basierten Gemeinsamen Gesundheitsakte. In 50. Jahrestagung der Deutschen Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie (gmds). Freiburg im Breisgau. http://www.egms.de/static/en/meetings/gmds2005/05gmds355.shtml [Stand 2010-03-28].

Bergmann, Joachim 2006. *Zur Architektur transinstitutioneller elektronischer Patientenakten zur Unterstützung vernetzter Versorgungsstrukturen*. Dissertation. Technische Universität Braunschweig, Braunschweig.

Bien, Adam 2009. *Real World Java EE Patterns Rethinking Best Practices*. Iteration One. press.adam-bien.com.

Blobel, Bernd 2008. EHR Architectures - Comparison and Trends. In *eHealth: Combining Health Telematics, Telemedicine, Biomedical Engineering and Bioinformatics to the Edge*. Studies in Health Technology and Informatics. Regensburg: IOS Press, 59–73.

Blobel, Bernd 2007. EPA-Modelle im Vergleich: openEHR, HL7 V3 Specs, EN/ISO 13606, CCR. In G. Steyer & T. Tolxdorff *TELEMED 2007 – Electronic Health Record und Gesundheitsportale*. Telemed 2007. Berlin.

BMI 2001. Registraturrichtlinie für das Bearbeiten und Verwalten von Schriftgut in Bundesministerien, Bundesministerium des Innern (Herausgeber), Stabsstelle Moderner Staat – Moderne Verwaltung. http://www.verwaltung-innovativ.de/cln_110/nn_684674/SharedDocs/Publikationen/Bestellservice/moderner__staat__registraturrichtlinie,templateId=raw,property=publicationFile.pdf/moderner_staat_registraturrichtlinie.pdf [Stand 2010-02-12].

Böckmann, Britta & Houta, Salima 2008. Entwicklung und Implementierung eines generellen Austauschformats zum Management intersektoraler Pfade auf Basis HL7 V3. In *Brückenschlag von Medizinischer Informatik, Biometrie und Epidemiologie zur Medizintechnik*. GMDS 2008. Stuttgart, 53–55.

Bodendorf, Freimut 2008. *Daten- und Wissensmanagement*. 2., aktualisierte u. erw. Aufl. Springer, Berlin.

Boeske, Martin u. a. 2004. Managementpapier „Elektronische Patientenakte“ - Aktionsforum Telematik im Gesundheitswesen. http://ehealth.gvg-koeln.de//cms/medium/676/MP_ePa_050124.pdf [Stand 2010-03-10].

Bointner, Karl & Duftschmied, Georg 2008. Semantische Interoperabilität im elektronischen Gesundheitsdatenaustausch mittels dualer Modellierung - Der HL7 Templates Ansatz. In *Medical Informatics meets eHealth*. eHealth 2008. Wien, 133–138. http://www.ehealth2011.at/archiv/eHealth2008/program/presentations/Postersession/bointner_paper.pdf [Stand 2011-01-8].

Breitling, Max 2001. *Formale Fehlermodellierung für verteilte reaktive Systeme*. Dissertation. Technische Universität München, München. http://tumb1.biblio.tu-muenchen.de/publ/diss/in/2001/breitling.pdf.

Brown, Aaron & Weihl, Bill 2011. *An update on Google Health and Google PowerMeter | Official Google Blog*. http://googleblog.blogspot.de/2011/06/update-on-google-health-and-google.html#!/2011/06/update-on-google-health-and-google.html [Stand 2012-08-27].

Brown, William J. u. a. 1998. *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis*. 1. Auflage. John Wiley & Sons.

Brunner, Manfred & Kaiser, Jochen 2008. Technologische Entwicklungen erfordern die Integration neuer Kompetenzen in den Datenschutzprozess am Krankenhaus. In *Brückenschlag von Medizinischer Informatik, Biometrie und Epidemiologie zur Medizintechnik*. GMDS 2008. Stuttgart, 72–74.

Buschmann, Frank u. a. 1996. *Pattern-Oriented Software Architecture 1: A System of Patterns*. 1. Auflage. Wiley & Sons.

Buschmann, Frank, Henney, Kevlin & Schmidt, Douglas C. 2007a. *Pattern-Oriented Software Architecture 4: A Pattern Language for Distributed Computing*. 1. Auflage. Wiley & Sons.

Buschmann, Frank, Henney, Kevlin & Schmidt, Douglas C. 2007b. *Pattern-Oriented Software Architecture 5: On Patterns and Pattern Languages*. 1. Auflage. Wiley & Sons.

Chen, Roland S. u. a. 2000. Exploring Performance Issues for a Clinical Database Organized Using an Entity-Attribute-Value Representation. *J Am Med Inform Assoc. 2000 Sep-Oct* 7, 5, 475–487.

Choi, Y.B. u. a. 2006. Telemedicine in the USA: standardization through information management and technical applications. *Communications Magazine, IEEE* 44, 4, 41–48.

Clement, Reiner & Schreiber, Dirk 2010. IT-Standards, Lock-In und Switching-costs. In *Internet-Ökonomie*. Physica-Lehrbuch. Physica-Verlag, 209–243. http://dx.doi.org/10.1007/978-3-7908-2596-1_8.

Cockburn, Alistair 2008. *Use Cases effektiv erstellen*. 1. Nachdruck. Mitp-Verlag.

Cockburn, Alistair 2000. *Writing Effective Use Cases*. Addison-Wesley Professional.

Coplien, James O. 1991. *Advanced C++ Programming Styles and Idioms*. Addison-Wesley Longman, Amsterdam.

Coulouris, G., Dollimore, J. & Kindberg, T. 2001. *Verteilte Systeme: Konzepte und Design*. 3., überarbeitete Auflage. Pearson Education.

Delessy, N., Fernandez, E.B. & Larrondo-Petrie, M.M. 2007. A Pattern Language for Identity Management. In International Multi-Conference on Computing in the Global Information Technology, 2007. ICCGI 2007. Guadeloupe, French Caribbean.

Delessy, Nelly u. a. 2007. Patterns for access control in distributed systems. In *Proceedings of the 14th Conference on Pattern Languages of Programs*. Monticello, Illinois: ACM, 1–11.

Demski, Hans, Waegemann, Peter & Engelbrecht, Rolf 2008. Der amerikanische CCR Standard - Impulse für Kontinuität in der medizinischen Dokumentation. In *Brückenschlag von Medizinischer Informatik, Biometrie und Epidemiologie zur Medizintechnik*. GMDS 2008. Stuttgart. http://www.egms.de/static/en/meetings/gmds2008/08gmds167.shtml [Stand 2011-01-10].

Dierks, Christian 1999. *Rechtliche und praktische Probleme der Integration von Telemedizin in das Gesundheitswesen in Deutschland*. Habilitation. Humboldt-Universität zu Berlin, . http://deposit.ddb.de/cgi-bin/dokserv?idn=959449531&dok_var=d1&dok_ext=pdf&filename=959449531.pdf [Stand 2012-02-10].

Dimakopoulos, Vassilios V. & Pitoura, Evaggelia 2006. On the Performance of Flooding-Based Resource Discovery. *IEEE Trans. Parallel Distrib. Syst.* 17, 11, 1242–1252. http://portal.acm.org/citation.cfm?id=1175890.1176139 [Stand 2010-04-9].

Dolin, Robert H. u. a. 2004. HL7 Clinical Document Architecture, Release 2.0. http://hl7.org/library/Committees/structure/CDA.ReleaseTwo.CommitteeBallot03.Aug.2004.zip [Stand 2011-01-27].

Duennebeil, Sebastian u. a. 2009. Integration of Patient Health Portals into the German Healthcare Telematics Infrastructure. In *AMCIS 2009 Proceedings*. Americas Conference on Information Systems (AMCIS). Paper 754. http://aisel.aisnet.org/amcis2009/754/.

Eckenbach, Mandy & Böckmann, Britta 2008. DiPP - Digitale Pfade im Gesundheitsnetz Prosper. In S. Schug & U. Engelmann *TELEMED 2008 - Telematikunterstützung für neue Versorgunsformen - Konzepte für regionale Telematikprojekte*. Telemed 2008. Heidelberg.

Eckert, Claudia 2011. 10 Authentifikation. In *IT-Sicherheit*. Oldenbourg Wissenschaftsverlag GmbH, 429–533. http://www.oldenbourg-link.com/doi/book/10.1524/9783486595970.

Eggebraaten, Thomas J., Tenner, Jeffrey W. & Dubbels, Joel C. 2007. A health-care data model based on the HL7 reference information model. *IBM Syst. J.* 46, 1, 5–18.

Evans, Eric 2004. *Domain-driven design*. Addison-Wesley.

Fink, Andreas 2009. Monolithisches IT-System K. Kurbel u. a. *Enzyklopädie der Wirtschaftsinformatik* Online-Lexikon. http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Systementwicklung/Softwarearchitektur/Architekturparadigmen/Monolithisches-IT-System [Stand 2010-03-21].

Fowler, Martin 2002. *Patterns of Enterprise Application Architecture*. 1. Auflage. Addison-Wesley Professional.

Friedman, Charles P. & Wyatt, Jeremy 2006. *Evaluation methods in biomedical informatics*. Springer.

Gamma, Erich u. a. 2010. *Design Patterns*. 38. Auflage. Westford, Massachusetts: Addison-Wesley.

GI 2010. *Gesellschaft für Informatik e.V., Informatiklexikon, Stichwort: Software-Architektur*. http://www.gi-ev.de/no_cache/service/informatiklexikon/informatiklexikon-detailansicht/meldung/software-architektur-128.html [Stand 2010-03-12].

GI FG Softwaretechnik 2010. *Was ist Softwaretechnik?* http://pi.informatik.uni-siegen.de/gi/fg211/fg211_st_defs.html [Stand 2011-04-7].

Glesner, Sabine u. a. 2008. Manifest: Strategische Bedeutung des Software Engineering für die Medizin. *Computer Science-Research and Development* 22, 3, 127–135.

Google *Google Health Data API CCR Reference - Google Health Data API - Google Code*. http://code.google.com/intl/de-DE/apis/health/ccrg_reference.html [Stand 2011-01-20].

Graham, Ian 2003. *A Pattern Language for Web Usability*. 1st Auflage. Pearson Education.

Gross, Daniel & Yu, Eric 2001. From Non-Functional Requirements to Design through Patterns. *Requirements Engineering* Volume 6, Number 1, 18–36. http://www.springerlink.com/content/luclrv4jlc5wye4p/?MUD=MP [Stand 2012-09-29].

Gulliford, Martin, Naithani, Smriti & Morgan, Myfanwy 2006. What is „continuity of care“? *J Health Serv Res Policy* 11, 4, 248–250. http://jhsrp.rsmjournals.com/cgi/content/abstract/11/4/248 [Stand 2009-10-1].

Haas, Peter 2006. *Gesundheitstelematik*. Berlin [u.a.]: Springer.

Haas, Peter 2009. *Medizinische Informationssysteme und Elektronische Krankenakten*. 1. Auflage. Berlin: Springer.

Haggerty, Jeannie L u. a. 2003. Continuity of care: a multidisciplinary review. *BMJ* 327, 7425, 1219–1221. http://www.bmj.com [Stand 2009-10-1].

Hanmer, Robert 2007. *Patterns for Fault Tolerant Software*. 1. Auflage. John Wiley & Sons.

Hapner, Mark u. a. 2003. *Java Message Service*. http://jcp.org/en/jsr/detail?id=914.

Hasselbring, Wilhelm 2000. Information system integration. *Commun. ACM* 43, 6, 32–38. http://dx.doi.org/10.1145/336460.336472.

Hasselbring, Wilhelm & Ziesche, Peter 1997. Einsatz von Entwurfsmustern bei der Entwicklung eines föderierten Krankenhausinformationssystems mit CORBA. In Fachtagung Verteilte Objekte in Organisationen. Bamberg. ftp://ls10-ftp.cs.uni-dortmund.de/.../Hasselbring-Ziesche_SWT-Memo-92.ps.gz [Stand 2010-03-18].

HEAL NY Phase 5 Health IT & Public Health Team 2008. *Health Information Exchange (HIE) for Public Health Use Case*. New York: New York State - Department of Health. http://www.health.state.ny.us/technology/projects/docs/health_information_exchange_for_public_health_-_use_case.pdf.

Heineman, George T. & Councill, William T. 2001. *Component-Based Software Engineering: Putting the Pieces Together*. Addison-Wesley Longman, Amsterdam.

Hellmann, Wolfgang 2003. *Praxis Klinischer Pfade*. Ecomed.

Henney, Kevlin, Schlossberg, Edwin & Kelly, Allan 2005. Context encapsulation - three stories, a language, and some sequences. *IN PROCEEDINGS OF EUROPLOP 2005, IRSEE* . http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.97.1614 [Stand 2011-06-26].

Hen-Tov, A. u. a. 2009. ModelTalk: When Everything Is a Domain-Specific Language. *Software, IEEE* 26, 4, 39–46.

HL7 *About Health Level Seven International*. http://www.hl7.org/about/index.cfm [Stand 2011-01-13].

HL7 2011. *ANSI Approved Standards*. http://www.hl7.org/implement/standards/ansiapproved.cfm?ref=nav [Stand 2011-01-13].

HL7 2010. *What is the Continuity of Care Document?* http://wiki.hl7.org/index.php?title=Product_CCD#What_is_the_Continuity_of_Care_Document.3F [Stand 2011-01-20].

Hoerbst, Alexander u. a. 2009. Requirements regarding quality certification of Electronic Health Records. *Studies in Health Technology and Informatics* 150, 384–388. http://www.ncbi.nlm.nih.gov/pubmed/19745337 [Stand 2010-03-18].

Hohpe, Gregor & Woolf, Bobby 2010. *Enterprise integration patterns, Designing, Building, and Deploying Messaging Solutions*. 14th Printing. Addison-Wesley Professional.

IEEE 2005. IEEE Std. 1220-2005 - IEEE Standard for Application and Management of the Systems engineering Process. http://ieeexplore.ieee.org/servlet/opac?punumber=10106 [Stand 2010-03-21].

IETF 2010. *Smime Status Pages*. http://tools.ietf.org/wg/smime/ [Stand 2012-08-27].

IHTSDO *About SNOMED CT*. International Health Terminology Standards Development Organisation. http://www.ihtsdo.org/snomed-ct/snomed-ct0/ [Stand 2011-01-29].

IHTSDO 2009. SNOMED Clinical Terms® - User Guide, International Health Terminology Standards Development Organisation. http://www.ihtsdo.org/fileadmin/user_upload/Docs_01/SNOMED_CT/About_SNOMED_CT/Use_of_SNOMED_CT/SNOMED_CT_User_Guide_20090731.pdf [Stand 2011-01-29].

Imamura, Takeshi, Dillaway, Blair & Simon, Ed 2002. XML Encryption Syntax and Processing. http://www.w3.org/TR/xmlenc-core/ [Stand 2011-01-14].

Information Sciences Institute University of Southern California 1981. RFC 793 - Transmission Control Protocol, Internet Engineering Task Force (IETF). http://www.ietf.org/rfc/rfc793.txt [Stand 2013-02-11].

Ingram, David 2002. *openEHR - Origins of openEHR*. http://www.openehr.org/about/origins [Stand 2013-03-22].

Iseger, Martijn 2010. *Domain-specific modeling for generative software development*. http://www.developerfusion.com/article/84844/domainspecific-modeling-for-generative-software-development/ [Stand 2013-02-11].

Jacobson, Ivar 1997. *Object oriented software engineering*. Harlow, England [u.a.]: Addison-Wesley.

Jähn, Karl & Nagel, Eckhard 2004. *e-Health*. Heidelberg: Springer.

Jobst, Fritz 2001. *Programmieren in Java*. 3., überarbeitete Auflage. München, Wien: Carl Hanser Verlag.

Johnson, Ralph E. 1992. Documenting frameworks using patterns. In *conference proceedings on Object-oriented programming systems, languages, and applications*. Vancouver, British Columbia, Canada: ACM, 63–76. http://portal.acm.org/citation.cfm?id=141936.141943 [Stand 2009-10-14].

Josuttis, Nicolai M. 2007. *SOA in Practice: The Art of Distributed System Design*. 1. Auflage. O'Reilly Media.

Kahla-Witzsch, Heike Anette & Geisinger, Thomas 2004. *Clinical Pathways in der Krankenhauspraxis*. Kohlhammer.

Kammerer, F J, Frankewitsch, T & Prokosch, H-U 2009. Design of a web portal for interdisciplinary image retrieval from multiple online image resources. *Methods of Information in Medicine* 48, 4, 361–370. http://www.ncbi.nlm.nih.gov/pubmed/19448884 [Stand 2010-03-18].

Kaspar, Mathias 2005. *Analyse von Werkzeugen zur CDA-Erstellung*. Bachelor Arbeit. Georg-August-Universität Göttingen, . http://www.informatik.uni-goettingen.de/Filepool/Theses/gaug-zfi-bm-2005-31.pdf.

Kassenärztliche Bundesvereinigung 2006. *KBV - IT in der Arztpraxis - Schnittstellen - Weitere Informationen zu xDT*. http://www.kbv.de/ita/4274.html [Stand 2011-01-12].

Kassenärztliche Bundesvereinigung 2010. LDT Labordatenträger - Datensatzbeschreibung zur Übertragung von Laborberichten und -aufträgen - LDT1001.02 Version 4.13.

http://daris.kbv.de/daris/doccontent.dll?LibraryName=EXTDARIS^DMSSLAVE&SystemType=2&LogonId=39cccb76b6056db4f0d3681bf4dd7a9c&DocId=003734736&Page=1 [Stand 2011-01-13].

Kelly, S. & Pohjonen, R. 2009. Worst Practices for Domain-Specific Modeling. *Software, IEEE* 26, 4, 22–29.

Kilov, H. 1990. From semantic to object-oriented data modeling. In *Systems Integration, 1990. Systems Integration '90., Proceedings of the First International Conference on*. Systems Integration, 1990. Systems Integration '90., Proceedings of the First International Conference on. 385–393. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=138704&isnumber=3754.

Kitchenham, Barbara A., Dyba, Tore & Jorgensen, Magne 2004. Evidence-Based Software Engineering. In *Proceedings of the 26th International Conference on Software Engineering*. IEEE Computer Society, 273–281.

Kneubühl, Thomas u. a. 2009. *PatternFinder Konzept*. Masterarbeit. Uni Basel, Fachhochschule Nordwestschweiz und Hochschule für Technik Rapperswil, . http://eprints3.hsr.ch/8/1/PatternFinder.pdf [Stand 2012-05-4].

Kraus, Stefan 2008. *DQL: DICOM Query Language - Serviceorientierter Zugriff auf medizinische Bildarchive mittels domänenspezifischer Abfragesprache*. Diplomarbeit. Fachhochschule Regensburg, . http://www.opus-bayern.de/fh-regensburg/volltexte/2008/33/ [Stand 2009-10-9].

Kraus, Stefan u. a. 2008. Vereinfachung von DICOM-Querys durch eine domänenspezifische Abfragesprache. In *Brückenschlag von Medizinischer Informatik, Biometrie und Epidemiologie zur Medizintechnik*. GMDS 2008. Stuttgart, 329–331. http://www.egms.de/de/meetings/gmds2008/08gmds201.shtml [Stand 2009-03-5].

Krüger, Stefan 2008. *CLINICAL PATHWAY Behandlungspfad Pneumonie (CAP), Universitätsklinikum Aachen*. http://www.ukaachen.de/go/show?ID=6199238&DV=0&COMP=download&NAVID=21229134&NAVDV=0 [Stand 2013-01-27].

Kuchenbecker, J. & Behrens-Baumann, W. 2004. Einsatz einer elektronischen Patientenakte (EPA) an der Universitätsaugenklinik Magdeburg. *Der Ophthalmologe* 101, 12, 1214–1219.

Kurose, James F. & Ross, Keith W. 2008. *Computernetzwerke*. Pearson Education.

Lackes, Richard & Siepermann, Markus Stichwort: Softwaresystem. *Gabler Wirtschaftslexikon* online im Internet. http://wirtschaftslexikon.gabler.de/Archiv/57353/softwaresystem-v5.html [Stand 2010-03-5].

Lassmann, Wolfgang 2006. *Wirtschaftsinformatik*. Gabler Verlag.

Leiner, Florian u. a. 1999. *Medizinische Dokumentation*. 3. Auflage. Stuttgart [u.a.]: Schattauer.

Lenz, Richard & Kuhn, Klaus A. 2004. Aspekte einer prozessorientierten Systemarchitektur für Informationssysteme im Gesundheitswesen P. Dadam & M. Reichert. *Informatik 2004* Band 2, 530–536. http://subs.emis.de/LNI/Proceedings/Proceedings51.html.

Lloyd, David 1995. Good European Health Record (EC AIM GEHR A2014), Deliverable 19 - GEHR Architecture Version 1.0. http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/EUCEN/del19.pdf [Stand 2010-12-17].

Ludewig, Jochen & Lichter, Horst 2010. *Software Engineering*. 2., überarbeitete, aktualisierte und ergänzte Auflage. Heidelberg: dpunkt.-Verl.

Malich, Stefan 2008. Pattern im Kontext des Entwurfs und der Bewertung von Softwarearchitekturen. In *Qualität von Softwaresystemen*. 103–138. http://dx.doi.org/10.1007/978-3-8349-9726-5_4.

Mayr, Margit 2010. Weg vom Inseldenken – hin zum Denken in Systemen. *ProCare* 15, 1-2, 32–35. http://dx.doi.org/10.1007/s00735-009-0263-5.

McDonald, Clem u. a. 2010. *Logical Observation Identifiers Names and Codes (LOINC) - User' Guide*. http://loinc.org/downloads/files/LOINCManual.pdf [Stand 2011-02-4].

McIlroy, M. Douglas 1968. Mass Produced Software Components. In *Software Engineering, Report on a conference sponsored by the NATO Science Committee*. Garmisch, 138–155. http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF [Stand 2010-03-3].

Meszaros, Gerard & Doble, Jim 1997. A pattern language for pattern writing. In *Pattern languages of program design 3*. Addison-Wesley Longman Publishing Co., Inc., 529–574. http://portal.acm.org/citation.cfm?id=273487 [Stand 2008-07-22].

Microsoft *Message Queuing (MSMQ)*. http://msdn.microsoft.com/en-us/library/ms711472(v=vs.85).aspx [Stand 2013-a-03-15].

Microsoft *Storing CCR and CCD Data in HealthVault*. http://msdn.microsoft.com/en-us/healthvault/bb968871 [Stand 2011-b-01-20].

Misoch, Sabina 2006. *Online-Kommunikation*. Konstanz: UTB.

Morrison, Patrick & Fernandez, Eduardo B. 2006a. The Credential Pattern (Conference Version). In PLoP '06. Portland, Oregon, 1–8. http://hillside.net/plop/2006/Papers/Library/PLoP2006_Credential.pdf [Stand 2012-03-16].

Morrison, Patrick & Fernandez, Eduardo B. 2006b. The credentials pattern (ACM Version). In *Proceedings of the 2006 conference on Pattern languages of programs*. PLoP '06. Portland, Oregon: ACM, 1–4. http://portal.acm.org/citation.cfm?id=1415483 [Stand 2009-06-22].

Nakakoji, Kumiyo 2008. Supporting Software Development as Collective Creative Knowledge Work. In *Proceedings of the 2nd International Workshop on Supporting Knowledge Collaboration in Software Development*. 21st IEEE/ACM International Conference on Automated Software Engineering. Tokyo: National Institute of

Informatics (Tokyo, Japan), 1–8. http://se.naist.jp/achieve/pdf/376.pdf#page=9 [Stand 2011-04-22].

NEMA 2009a. Digital Imaging and Communications in Medicine (DICOM), Part 10: Media Storage and File Format for Media Interchange, National Electrical Manufacturers Association. ftp://medical.nema.org/medical/dicom/2009/09_10pu.pdf [Stand 2010-12-10].

NEMA 2009b. Digital Imaging and Communications in Medicine (DICOM), Part 18: Web Access to DICOM Persistent Objects (WADO), National Electrical Manufacturers Association. ftp://medical.nema.org/medical/dicom/2009/09_18pu.pdf [Stand 2011-01-9].

NEMA 2009c. Digital Imaging and Communications in Medicine (DICOM), Part 8: Network Communication Support for Message Exchange, National Electrical Manufacturers Association. ftp://medical.nema.org/medical/dicom/2009/09_08pu.pdf [Stand 2010-12-10].

Nilsson, Jimmy 2006. *Applying domain-driven design and patterns*. Addison-Wesley.

Ochsenschläger, Peter, Rieke, Roland & Velikova, Zaharina 2008. Die elektronische Krankenakte - Eine Sicherheitsstrategie. In *D-A-CH Security 2008. Proceedings: Bestandsaufnahme, Konzepte, Anwendungen, Perspektiven*. Syssec 2008. Basel.

Oemig, Frank 2006. HL7-v2.5-Nachrichtenprofil: ADT-Profil zur Patientenaufnahme. http://www.hl7.de/download/documents/profile_2.0/Aufnahme_v2.0.pdf [Stand 2011-01-13].

Oestereich, Bernd 2009. *Analyse und Design mit UML 2.3: Objektorientierte Softwareentwicklung*. 9., aktualisierte und erweiterte Auflage. Oldenbourg.

OMG 2009. OMG Unified Modeling LanguageTM (OMG UML), Superstructure, Version 2.2, Object Management Group. http://www.omg.org/spec/UML/2.2/Superstructure.

openEHR 2008. *GeHR in Australia - The Good electronic Health Record - openEHR :: future proof and flexible EHR specifications*. http://www.openehr.org/shared-resources/gehr_all/gehr_australia.html [Stand 2009-07-31].

Oracle *Java Remote Method Invocation - Distributed Computing for Java*. http://www.oracle.com/technetwork/java/javase/tech/index-jsp-138781.html [Stand 2013-03-22].

Partsch, Helmuth 2010. *Requirements-engineering systematisch*. Springer.

Pink, Axel u. a. 2002. *Software-Entwicklung für Kommunikationsnetze*. Springer.

Portland Pattern Repository 2009. *Handle Body Pattern*. http://c2.com/cgi/wiki?HandleBodyPattern [Stand 2013-03-15].

Prechelt, Lutz u. a. 2001. A Controlled Experiment in Maintenance Comparing Design Patterns to Simpler Solutions. *IEEE Transactions on Software Engineering* No. 12, Vol. 27, 1134–1144. http://www.computer.org/portal/web/csdl/doi/10.1109/32.988711.

Prechelt, Lutz u. a. 2002. Two controlled experiments assessing the usefulness of design pattern documentation in program maintenance. *IEEE Transactions on Software Engineering* Volume 28, Issue 6, 595–606.

Priebe, Torsten u. a. 2004. A Pattern System for Access Control C. Farkas & P. Samarati. *Research Directions in Data and Applications Security XVIII* 144, 235–249. http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.114.7637.

Priebe, Torsten u. a. 2005. ABAC - Ein Referenzmodell für attributbasierte Zugriffskontrolle. In H. Federrath *Sicherheit 2005*. LNI. 285–296. http://subs.emis.de/LNI/Proceedings/Proceedings62/GI-Proceedings.62-30.pdf.

Prokosch, Hans-Ulrich 2001. KAS, KIS, EKA, EPA, EGA, E-Health: Ein Plädoyer gegen die babylonische Begriffsverwirrung in der Medizinischen Informatik. *Informatik, Biometrie und Epidemiologie in Medizin und Biologie* 32, 4, 371–382. http://www.imi.med.uni-erlangen.de/fileadmin/Forschung/Publikationen_pdf/ProkoschHU.2001.KAS-KIS-EKA.pdf [Stand 2013-02-12].

Pschichholz, Holger u. a. 2008. Freiburger Einweiserportal: Terminbuchung, Befundzugriff, Informationsplattform. In S. Schug & U. Engelmann *TELEMED 2008 - Telematikunterstützung für neue Versorgunsformen - Konzepte für regionale Telematikprojekte*. Telemed 2008. Heidelberg, 72–77.

Queensland Health Clinical Pathways Board 2002. *Homepage | Clinical Pathways | Patient Safety and Quality Improvement Service*. http://www.health.qld.gov.au/psq/pathways/default.asp [Stand 2013-02-13].

Ramsdell, Blake & Turner, Sean 2010. RFC 5751 - Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification, Internet Engineering Task Force (IETF). http://tools.ietf.org/html/rfc5751 [Stand 2012-08-27].

Ratzka, Andreas 2010. *Patternbasiertes User Interface Design für multimodale Interaktion: Identifikation und Validierung von Patterns auf Basis einer Analyse der Forschungsliteratur und explorativer Benutzertests an Systemprototypen*. 1. Auflage. Hülsbusch.

Rechenberg, Peter 2000. *Was ist Informatik? : eine allgemeinverständliche Einführung*. Hanser Verlag.

Riebling, Jürgen 2007. Intersektorale Kommunikation zwischen Praxis und Klinik. In G. Steyer & T. Tolxdorff *TELEMED 2007 – Electronic Health Record und Gesundheitsportale*. Telemed 2007. Berlin.

Rienhoff, Otto 2004. Bedeutung der Kompetenznetze für die Innere Medizin. *Medizinische Klinik* 99, 7, 407–411. http://dx.doi.org/10.1007/s00063-004-1063-0 [Stand 2010-03-23].

Robertson, Suzanne & Robertson, James 2006. *Mastering the Requirements Process*. Pearson Education.

Salingaros, Nikos A. 2000. The Structure of Pattern Languages. *arq: Architectural Research Quarterly* 4, 02, 149–162. http://journals.cambridge.org/action/displayAbstract?fromPage=online&aid=1711496&fulltextType=RA&fileId=S1359135500002591 [Stand 2011-08-2].

Sametinger, Johannes 1997. *Software engineering with reusable components*. Berlin [u.a.]: Springer.

Sanchez, Cesar, Triana, Edwin & Romero, Eduardo 2008. A flexible web oriented telehealth platform using a RIM-HL7 based model. In *Proceedings of the 2008 Euro American Conference on Telematics and Information Systems*. EATIS '08. Aracaju, Brazil: ACM, 7:1–7:8.

Schadow, Gunther, Mead, Charles N. & Walker, D. Mead 2006. The HL7 Reference Information Model Under Scrutiny. *Stud Health Technol Inform* 124, 124–151. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.106.380 [Stand 2011-02-10].

Schadow, Gunther, Rishel, Wesley J. & Tucker, Mark C. 1999. Health Level Seven - Secure HL7 Transactions using Internet Mail, Version 1, Revision 1.7. http://www.hl7.org/Library/Committees/secure/HL7-Mime-1.7.pdf [Stand 2011-01-8].

Schobert, Wolfram J.H. 2005. *Ansätze zur Visualisierung von Pattern-Languages*. Diplomarbeit. FernUniversität Hagen, . http://www.pi6.fernuni-hagen.de/publ/abschlussarbeiten/schobert2005.pdf [Stand 2011-08-1].

Schramm-Wölk, Ingeborg & Schug, Stephan H. 2004. e-Patientenakte und e-Gesundheitsakte. In K. Jähn & E. Nagel *e-Health*. Heidelberg: Springer, 16–22.

Schuemmer, Till 2003. Seeking for structure in a Groupware Pattern Language. In *Workshop at CHI 2003*. CHI 2003. Fort Lauderdale. http://www.cs.kent.ac.uk/people/staff/saf/patterns/chi2003/submissions/till_Pattern%20Language%20Structure.pdf [Stand 2011-08-2].

Schug, Stephan H. & Schramm-Wölk, Ingeborg 2004. Telematik-Standards für das Gesundheitswesen. In K. Jähn & E. Nagel *e-Health*. Heidelberg: Springer, 11–15.

Schumacher, Markus u. a. 2005. *Security Patterns: Integrating Security and Systems Engineering*. 1. Auflage. Wiley & Sons.

Schwarze, J. C. u. a. 2005. Eine modulare Gesundheitsakte als Antwort auf Kommunikationsprobleme im Gesundheitswesen. *Wirtschaftsinformatik* 47, 3, 187–195.

Siddle, James 2011. „Choose your own architecture" - interactive pattern storytelling. In J. Noble u. a. *Transactions on pattern languages of programming II*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 16–33. http://dx.doi.org/10.1007/978-3-642-19432-0_2.

Siddle, James & Platts, Maisie 2009. *„Choose Your Own Architecture" - Interactive Pattern Storytelling*. http://www.jamessiddle.net/docs/cyoa.pdf [Stand 2011-06-26].

Smith, Barry & Ceusters, Werner 2006. HL7 RIM: an incoherent standard. *Studies in health technology and informatics* 124, 133–138. http://view.ncbi.nlm.nih.gov/pubmed/17108516 [Stand 2010-11-18].

Smith, Jason McC. 2011. The Pattern Instance Notation: A simple hierarchical visual notation for the dynamic visualization and comprehension of software patterns. *Journal of Visual Languages & Computing* Volume 22, Issue 5, 355–374. http://dx.doi.org/10.1016/j.jvlc.2011.03.003.

Smith, Jason McC. & Stotts, David 2002. Elemental Design Patterns: A Formal Semantics for Composition of OO Software Architecture. In *Proceedings of 27th Annual IEEE/NASA Software Engineering Workshop*. 183–190.

Smith, Jason McC. & Stotts, David 2003. SPQR: flexible automated design pattern extraction from source code. In 18th IEEE Int. Conf. on Automated Software Engineering. 215–224. http://dx.doi.org/10.1109/ASE.2003.1240309.

Sommerville, Ian 2007. *Software Engineering*. 8., aktualisierte Auflage. München [u.a.]: Pearson Studium.

Srinivasan, R. 1995. RFC 1831 - RPC: Remote Procedure Call Protocol Specification Version 2, Internet Engineering Task Force (IETF). http://tools.ietf.org/html/rfc1831 [Stand 2010-03-2].

Steinmetz, Ralf & Wehrle, Klaus 2004. Peer-to-Peer-Networking & -Computing. *Informatik-Spektrum* 27, 1, 51–54. http://dx.doi.org/10.1007/s00287-003-0362-9.

Sun Microsystems 1988. RFC 1057 - RPC: Remote Procedure Call Protocol specification: Version 2, Internet Engineering Task Force (IETF). http://tools.ietf.org/html/rfc1057 [Stand 2010-03-2].

Szostek, Agnieszka Matysiak & Markopoulos, Panos 2006. Factors defining face-to-face interruptions in the office environment. In *CHI '06 extended abstracts on Human factors in computing systems*. CHI 2006. Montreal, Canada: ACM, 1379–1384.

Tanenbaum, A. & Van Steen, M. 2003. *Verteilte Systeme - Grundlagen und Paradigmen*. Pearson Studium.

Tichy, Walter F. & Unger, Barbara 2000. Do Design Patterns Improve Communication? An Experiment with Pair Design. In *WESS 2000*. WESS 2000. http://www.ipd.uka.de/Tichy/uploads/publikationen/149/wess.pdf.

U.S. National Library of Medicine 2009. *SNOMED Clinical Terms® (SNOMED CT®)*. http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html [Stand 2011-01-29].

UAG KIS 2011. Orientierungshilfe Krankenhausinformationssysteme Unterarbeitsgruppe Krankenhausinformationssysteme der Arbeitskreise Gesundheit und Soziales sowie Technische und organisatorische Datenschutzfragen der Konferenz der Datenschutzbeauftragten des Bundes und der Länder. http://www.lfd.niedersachsen.de/download/57482 [Stand 2012-02-10].

Uwe Schneider & Dieter Werner 2007. *Taschenbuch der Informatik*. 6. Auflage. München: Fachbuchverlag Leipzig im Carl-Hanser-Verlag.

Volker Eric Amelung & Axel Mühlbacher Stichwort: elektronische Patienteakte. *Gabler Wirtschaftslexikon*. http://wirtschaftslexikon.gabler.de/Archiv/17946/elektronische-patientenakte-v6.html.

Volker Eric Amelung, Axel Mühlbacher & Christian Krauth Stichwort: Integrierte Versorgung. *Gabler Wirtschaftslexikon* Online im Internet. http://wirtschaftslexikon.gabler.de/Archiv/17861/integrierte-versorgung-v9.html [Stand 2010-03-12].

Vreeman, Daniel 2010. *LOINC Background — LOINC*. http://loinc.org/background [Stand 2011-02-4].

Waegemann, C. Peter 2003. EHR vs. CPR vs. EMR. *Healthcare Informatics Online* . http://www.providersedge.com/ehdocs/ehr_articles/EHR_vs_CPR_vs_EMR.pdf [Stand 2013-02-12].

Waegemann, C. Peter 2007. RHIOs und CCR – Wege zur elektronischen Patientenakte in den USA. In G. Steyer & T. Tolxdorff *TELEMED 2007 – Electronic Health Record und Gesundheitsportale*. TELEMED 2007. Berlin.

Van Welie, Martijn & Van der Veer, Gerrit C. 2003. Pattern Languages in Interaction Design: Structure and Organization. In M. Rauterberg, M. Menozzi, & J. Wesson *INTERACT*. Human-Computer Interaction INTERACT'03: IFIP TC13 International Conference on Human-Computer Interaction. Zürich: IOS Press, 527–534.

Wendroth, Lothar 2001. Terminalserver. *cms-journal Nr. 22* 25–26. http://edoc.hu-berlin.de/docviews/abstract.php?id=20235 [Stand 2010-04-7].

White, James E. 1976. A high-level framework for network-based resource sharing. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*. New York, New York: ACM, 561–570. http://dx.doi.org/10.1145/1499799.1499878.

WHO 2004. *ICD-10: International Statistical Classification of Diseases and Related Health Problems - Tenth Revision - Volume 2 - Second Edition*. Genf: World Health Organization. http://www.who.int/classifications/icd/ICD-10_2nd_ed_volume2.pdf [Stand 2011-01-28].

WHO *International Classification of Diseases (ICD)*. World Health Organization (WHO). http://www.who.int/classifications/icd/en/ [Stand 2011-01-28].

Wiedermann, Wolfgang, Tsakpinis, Athanassios & Wolff, Christian 2008. Generischer Architekturansatz für Telemedizin Portale und verteilte Krankenakten. In S. Schug & U. Engelmann *TELEMED 2008 - Telematikunterstützung für neue Versorgunsformen - Konzepte für regionale Telematikprojekte*. Telemed 2008. Heidelberg, 65–71.

Wiedermann, Wolfgang, Wolff, Christian & Tsakpinis, Athanassios 2009. Service oriented approach for multi backend retrieval in medical systems. In CBMS 2009. 22nd IEEE International Symposium on Computer-Based Medical Systems. 1–4. http://dx.doi.org/10.1109/CBMS.2009.5255450.

Withall, Stephen 2007. *Software Requirement Patterns*. Microsoft Press.

Wurhofer, Daniela u. a. 2009. Introducing a Comprehensive Quality Criteria Framework for Validating Patterns. In *Proceedings of the 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns*. IEEE Computer Society, 242–247.

Yoder, Joseph & Barcalow, Jeffrey 1998. Architectural Patterns for Enabling Application Security. In *Proceedings of the 4th Pattern Language of Programming Conference*. PLoP ´97. Monticello, Illinois, USA. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.2274 [Stand 2009-08-27].

Zilahi-Szabó, Miklós Géza 1995. *Kleines Lexikon der Informatik und Wirtschaftsinformatik*. München [u.a.]: Oldenbourg.

Zorneck, Christian 2009. *Untersuchung von Standards und Lösungen zur Online-Einweisung von Patienten*. Diplomarbeit. Fachhochschule Regensburg, Regensburg.