REBECCA REUTER

TECHNOLOGIEBASIERTE UNTERSTÜTZUNGSMASSNAHMEN IN DER AKADEMISCHEN SOFTWARE ENGINEERING-AUSBILDUNG

TECHNOLOGIEBASIERTE UNTERSTÜTZUNGSMASSNAHMEN IN DER AKADEMISCHEN SOFTWARE ENGINEERING-AUSBILDUNG

Konzeption, Entwicklung und Evaluation einer lernerzentrierten Analyse- und Designumgebung

Inaugural-Dissertation zur Erlangung der Doktorwürde
der Fakultät für Sprach-, Literatur- und Kulturwissenschaften der Universität Regensburg

vorgelegt von

Rebecca Reuter
aus
Regensburg
2020

Rebecca Reuter: *Technologiebasierte Unterstützungsmaßnahmen in der akademischen Software Engineering-Ausbildung*, Konzeption, Entwicklung und Evaluation einer lernerzentrierten Analyse- und Designumgebung, © 2020

ERSTGUTACHTER: Prof. Dr. Christian Wolff
ZWEITGUTACHTER: Prof. Dr. Jürgen Mottok

KOOPERATION: Die Arbeit entstand in gemeinsamer Betreuung durch die Fakultät für Sprach-, Literatur- und Kulturwissenschaften der Universität Regensburg und die OTH Regensburg.

ABSTRACT

The ability of model building, modeling poses not only a central, but also a very complex, task for software engineers. By means of abstraction, a model of the real world is created, which is supposed to help to understand problems. Teaching software modeling therefore becomes a challenging task for both students and instructors. This thesis describes an approach for the problem-based initiation of tools (so-called scaffolds) to support students in modeling with the Unified Modeling Language (UML). As a basis, problems encountered by students during modeling of a software system with the UML were recorded and cataloged. Based on the identified problems, various scaffolds were derived and conceptualized. These include documentation and tutorials as traditional tools as well as technology-based approaches such as the use of augmented reality, eye-movement modeling examples, and tools for identifying use cases as well as class candidates. The various scaffolds were prototyped and evaluated in a modeling environment. The evaluations with students and an expert review show promising results. The work confirms that evidence-based scaffolds and their integration into teaching applications can enrich university teaching in practice.

ZUSAMMENFASSUNG

Die Fähigkeit der Modellbildung, das Modellieren stellt nicht nur eine zentrale, sondern auch eine sehr komplexe, Aufgabe für Software-Ingenieure dar. Durch die Tätigkeit des Abstrahierens wird ein Modell der realen Welt erstellt, das dabei helfen soll Probleme zu verstehen. Die Lehre zur Modellierung von Softwaresystemen nimmt für Studierende wie Lehrende daher eine herausfordernde Bedeutung ein. Diese Arbeit beschreibt einen Ansatz zur problembasierten Initiierung von Hilfsmitteln (sog. Scaffolds) zur Unterstützung Studierender bei der Modellierung mit der Unified Modeling Language (UML). Als Ausgangsbasis wurden Probleme Studierender bei der Modellierung von Softwaresystemen mit der UML erfasst und katalogisiert. Auf Basis der identifizierten Probleme wurden verschiedene Scaffolds abgeleitet und konzeptioniert. Dazu gehören sowohl Unterlagen und Tutorials als klassische Hilfsmittel wie auch technologiebasierte Ansätze, wie die Verwendung von Augmented Reality, Eye-Movement Modeling Examples und Hilfsmittel zur Identifikation von Use-Cases sowie Klassenkandidaten. Die verschiedenen Scaffolds wurden in eine Modellierungsumgebung prototypisch integriert und evaluiert. Die Evaluationen mit Studierenden sowie eine Expertenbegutachtung zeigen erfolgsversprechende Ergebnisse. Die Arbeit bestätigt, dass evidenzbasierte Scaffolds und deren Integration in den Lehreinsatz die praktische Hochschuldidaktik bereichern können.

PUBLIKATIONEN

Dieses Kapitel listet Publikationen, an denen die Autorin beteiligt war. Einige Ideen und Abbildungen sind bereits in folgenden Publikationen beigetragen worden. Sofern eigene Publikationen in der Arbeit herangezogen werden, sind sie entsprechend markiert.

Beslmeisl, M., Reuter, R. & Mottok, J. (2017). The importance of writing in software engineering education. In *Advances in Intelligent Systems and Computing* (S. 315–321). doi:10.1007/978-3-319-50337-0_29

Hauser, F., Reuter, R., Gegenfurtner, A., Gruber, H., Mottok, J. M. & Hutzler, I. (2019). Heuristics in Software Modelling: An Eyetracking Study. *Earli Book of Abstracts*. Verfügbar unter https://earli.org/sites/default/files/2019-09/BOA-2019.pdf

Hauser, F., Reuter, R., Gegenfurtner, A., Gruber, H. G. & Mottok, J. (2019). Eye Movements in Software Modelling - What Do They Tell Us About Heuristics. *ICERI2019 Proceedings*, 6064–6070. doi:10.21125/iceri.2019.1469

Hutzler, I., Hauser, F., Reuter, R., Mottok, J. & Gruber, H. (2018). Will the Noun/Verb Analysis Be Used To Generate Class Diagrams? an Eye Tracking Study. *ICERI2018 Proceedings*, 505–514. doi:10.21125/iceri.2018.1103

Klopp, M., Gold-Veerkamp, C., Abke, J., Borgeest, K., Reuter, R., Jahn, S., … Landes, D. (2020). Totally Different and yet so Alike: Three Concepts to Use Scrum in Higher Education. *Association for Computing Machinery*, 12–21. doi:10.1145/3396802.3396817

Knietzsch, M., Muckelbauer, D., Reuter, R. & Mottok, J. (2018). Experimental Verification of Indicated Benefits of Integrating Augmented Reality Into Academic Software Engineering Classes. In L. G. Chova, A. L. Martínez & I. C. Torres (Hrsg.), *ICERI2018 Proceedings* (S. 243–252). doi:10.21125/iceri.2018.1057

Müller, L., Jahn, S., Reuter, R. & Mottok, J. (2018). A Task Design Concept for a Virtual Classroom for Requirements Engineering Education. In L. G. Chova, A. L. Martínez & I. C. Torres (Hrsg.), *ICERI2018 Proceedings* (S. 911–920). doi:10.21125/iceri.2018.1216

Reuter, R., Stark, T., Sedelmaier, Y., Landes, D., Mottok, J. & Wolff, C. (2020). Insights in Students' Problems during UML Modeling. *IEEE Global Engineering Education Conference, EDUCON*, 592–600.

Reuter, R., Beslmeisl, M. & Mottok, J. (2017). Work in progress: Teaching-obstacles in higher software engineering education. *IEEE Global Engineering Education Conference, EDUCON*, 1631–1635. doi:10.1109/EDUCON.2017.7943067

Reuter, R., Hauser, F., Gold-Veerkamp, C., Mottok, J. & Abke, J. (2017). Towards a Definition and Identification of Learning Obstacles in Higher Software Engineering Education. *EDULEARN17 Proceedings*, *1*, 10259–10267. doi:10.21125/edulearn.2017.0943

Reuter, R., Hauser, F., Gold-Veerkamp, C., Stark, T., Kis, J., Mottok, J., … Meyer, D. (2018). Towards the construction of a questionnaire for the identification of learning obstacles. *IEEE Global Engineering Education Conference, EDUCON*, *2018-April*, 457–466. doi:10.1109/EDUCON.2018.8363266

Reuter, R., Hauser, F., Muckelbauer, D., Stark, T., Antoni, E., Mottok, J. & Wolff, C. (2019). Using Augmented Reality in Software Engineering Education? First insights to a comparative study of 2D and AR UML modeling. *Proceedings of the 52nd Hawaii International Conference on System Sciences CC BY-NC-ND 4.0*, *6*, 7798–7807. doi:10.24251/hicss.2019.938

Reuter, R., Jahn, S., Figas, P., Bartel, A., Mottok, J. & Hagel, G. (2018). Learning Tasks for Software Engineering Education. In *ACM European Conference on Software Engineering Education (ECSEE)* (S. 1–7). doi:10.1145/3209087.3209097

Reuter, R., Knietzsch, M., Hauser, F. & Mottok, J. (2019). Supporting abstraction skills using augmented reality? *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*. doi:10.1145/3304221.3325562

Reuter, R., Kuhn, M. & Mottok, J. (2018). A Two-Sided Approach of Applying Software Engineering Perspectives in Higher Education. *Advances in Intelligent Systems and Computing*, *715*, 460–467. doi:10.1007/978-3-319-73210-7_55

Reuter, R., Langer, T., Hauser, F., Muckelbauer, D., Gegenfurtner, A. & Mottok, J. (2019). Automatic generation of areas of interest in eye tracking: The case of software engineering. *Earli Book of Abstracts*. Verfügbar unter https://earli.org/sites/default/files/2019-09/BOA-2019.pdf

Reuter, R. & Mottok, J. (2016). Extending the Family of Inductive Teaching and Learning Methods Agile Teaching and Learning as Feature or Method ? In *Proceedings of the International Conference on Software Engineering Education*, Seeon: Shaker.

Soska, A., Reuter, R., Hauser, F., Reiß, M. & Mottok, J. (2017). Scaffolding in der Lehre von Design Pattern. *Tagungsband zum 3. Symposium zur Hochschullehre in den MINT-Fächern*, 112–116.

DANKSAGUNG

Im Entstehungsprozess dieser Arbeit habe ich viele Facetten des Promovierens kennengelernt. An dieser Stelle sei Allen, die mich in diesem Prozess in unterschiedlicher Form begleitet und unterstützt haben, von ganzem Herzen gedankt.

Mein ganz besonderer Dank gilt meinen Betreuern und Gutachtern Prof. Dr. Jürgen Mottok und Prof. Dr. Christian Wolff. Prof. Dr. Jürgen Mottok danke ich für die Freiheiten im Rahmen der Forschungstätigkeit und die unzähligen visionären Gedanken und bereichernden Diskussionen über die gesamte Zeit der Promotion hinweg. Prof. Dr. Christian Wolff möchte ich für die wertschätzende und wertvolle Unterstützung danken und die, im Nachhinein betrachtet, sehr vorausahnenden, wegweisenden Hinweise ab dem ersten Gespräch bis zur Finalisierung der Arbeit. Ebenso bedanke ich mich bei Prof. Dr. Andreas Gegenfurtner für die Begutachtung dieser Arbeit.

Darüber hinaus möchte ich mich bei all jenen Personen bedanken, die die Arbeit teilweise oder vollständig gelesen haben, mit ihren Gedanken, Anmerkungen sowie Unterstützung zu Erhebungen und Auswertungen die Arbeit bereichert haben, allen voran Theresa Stark, Juliane Kis und Sabrina Jahn. Theresa Stark gilt zudem mein ganz besonderer Dank für die jederzeit wertvolle Unterstützung in allen Phasen meiner Promotionszeit. Stefan Schreistetter, Florian Hauser und Dr. Kenneth Holmqvist sei gedankt für die Beratungen zur Durchführung und Auswertung der Eyetracking-Studie. Danken möchte ich ebenso Daniel Muckelbauer, Marco Knietzsch und Leonie Müller für Beiträge zur Softwareentwicklung und Unterstützung bei Erhebungen. Dankbar bin ich ebenso meinen Kolleginnen und Kollegen des LaS^3 und des Verbundprojekts Evelin. Ihr seid ein super Team. Allen Teilnehmerinnen und Teilnehmern der Universität Regensburg, der OTH Regensburg und der Hochschule Coburg sei für ihre Zeit in durchgeführten Studien und ihren Input gedankt.

Ganz besonders möchte ich meiner Familie danken. Meiner Mama, meiner Schwester und Tobias, für die jederzeit verständnisvolle und liebevolle seelische und moralische Unterstützung in den Höhen und Tiefen dieser Arbeit. Mein größter Dank gilt meinem Papa, ohne dessen Unterstützung über den gesamten Entstehungsprozess und unzähligen Korrekturvorschlägen die Arbeit sicherlich nicht die Qualität hätte, die sie heute hat. Weiterhin danke ich all meinen Freunden für ihre Geduld während dieser Zeit, für ihr Verständnis und diverse verkürzte Abende.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

QUELLCODEVERZEICHNIS

1

EINFÜHRUNG & MOTIVATION DER ARBEIT

You have to understand the problem.
What is the unknown?
What are the data?
What is the condition?
Is it possible to satisfy the condition?
Is the condition sufficient to determine the unknown?
Or is it insufficient? Or redundant? Or contradictory?
Draw a figure. Introduce suitable notation. Separate the various parts of the condition. Can you write them down?

— George Pòlya (Polya, 1985)

Mit diesen Worten leitet George Pòlya bereits 1945 sein Werk „How to solve it“ mit seinem deutschen Titel „Schule des Denkens“ ein. Sein Werk soll Lehrenden helfen, Studierenden Motive und Vorgehensweisen für die Lösung eines Problems aufzuzeigen, aber auch Studierende dabei unterstützen Wege zu finden, wie sie komplexe Aufgaben eigenständig angehen und adäquat lösen können. Zentral für seine Arbeit steht ein vierstufiges Konzept zum Lösen komplexer Aufgaben:

„**Erstens**
Du musst die Aufgabe verstehen
Zweitens
Suche den Zusammenhang zwischen den Daten und der Unbekannten
Du musst vielleicht Hilfsaufgaben betrachten, wenn ein unmittelbarer Zusammenhang nicht gefunden werden kann
Du musst schließlich einen Plan der Lösung erhalten
Drittens
Führe deinen Plan aus
Viertens
Prüfe die erhaltene Lösung“ (Polya, 2010, Deckblatt)

Für dieses Dissertationsvorhaben bedeuten das Werk und die zentralen Fragestellungen und gerade die Handlungsempfehlung zum Lösen einer Aufgabe viel mehr. Die Frage „How to solve it?“ oder wörtlich übersetzt „Wie kann man es lösen?“ prägt auch die Disziplin des Software Engineering, in der diese Arbeit angesiedelt ist. Software Engineering beschäftigt sich mit Prozessen, Methoden und Werkzeugen, um Software zu erstellen und zu pflegen und ist geprägt durch diese Frage.

Gerade in der Disziplin des Software Engineerings ist es für zukünftige Software-Ingenieure von entscheidender Bedeutung, Fähigkeiten zu entwickeln, die es ihnen ermöglichen, sich neues Wissen anzueignen, bereits erlerntes Wissen in unerwarteten Situationen anzuwenden und dynamische Probleme zu lösen. Kernkompetenzen eines Absolventen sollten neben der Problemlösekompetenz, die Fähigkeit sein, komplexe Prozesse zu verstehen, ein hohes Abstraktionsvermögen zu erwerben und eben auch Lernbereitschaft und Offenheit für Neues bereit zu halten (Sedelmaier, Claren & Landes, 2013, S. 119).

> „Eine der wichtigsten Aufgaben des Lehrers ist es, seinen Schülern zu helfen. Diese Aufgabe ist nicht ganz einfach; es erfordert Zeit, Übung, Hingabe und solide Prinzipien. Der Student sollte so viel Erfahrung in der selbständigen Arbeit wie möglich sammeln. Aber wenn er allein gelassen wird mit seinem Problem ohne oder mit unzureichender Hilfe, kommt er vielleicht überhaupt nicht voran. Wenn der Lehrer zu viel hilft, bleibt dem Schüler nichts mehr übrig. Der Lehrer sollte helfen, aber nicht zu viel und nicht zu wenig, damit der Schüler einen angemessenen Anteil an der Arbeit hat“ (Polya, 1985, S.1)

Nach Pòlya, kommt Dozierenden die schwierige Aufgabe zu, Studierenden in geeigneter Weise zu helfen und sie im Lernprozess, sowohl fachlich wie auch überfachlich, in der Aneignung der nötigen Kompetenzen zu unterstützen. Dabei stehen Dozierende vor der Herausforderung, so viel Hilfe wie nötig und so wenig Hilfestellung wie möglich zu geben, um ein möglichst selbstständiges Lernen zu fördern. Gerade bei Problemen sollten Studierende aber nicht alleine gelassen, sondern mit entsprechenden Hilfsmitteln unterstützt werden.

Die Annahme dieser Arbeit ist es, dass die Aussagen Pòlyas, auf die Disziplin des Software Engineerings und deren Lehre übertragbar sind und damit als zentrale Leitgedanken für die zu Grunde liegende Arbeit dienen können.

Wie in allen Ausbildungsformen geht es in der Software Engineering-Ausbildung an Hochschulen darum, ein Verhältnis zwischen Studieren-

dem, Lehrendem, der Aufgabe und den Hilfestellungen zu schaffen, um einen erfolgreichen Lernprozess zu erreichen. Wie jede Disziplin bringt auch Software Engineering domänenspezifische Eigenheiten mit sich, die Schwierigkeiten und Lernhindernisse beim Einstieg verursachen können. Beispielsweise wird häufig argumentiert, dass Software Engineering von Natur aus abstrakt ist, weshalb ein hohes Abstraktionsvermögen von Nöten ist (siehe z. B. Balzert & Balzert, 2009; Colburn & Shute, 2007; Dörge, 2015; Hartmann, Näf & Reichert, 2006). Besonders deutlich wird dies bei der Modellierung eines Softwaresystems. Schon der Terminus Modellbildung enthält dies: Es geht es darum, sich ein Bild der Aufgabenstellung zu machen, um diese zu verstehen, ein Modell, d. h. ein Modell der Aufgabenstellung zu entwerfen. Modellbildung gehört dabei zu den grundlegenden kognitiven Leistungen eines Menschen (Hesse & Mayr, 2008, S. 378) und ist nach Pòlyas Verständnis („Draw a figure“ (s.o.)) zentral für das Lösen einer Aufgabe (z. B. Polya, 2010, Deckblatt Schritt 2; Polya, 2010, S.19).

Ein Modell[1] ist demnach ein Repräsentant für etwas, das verstanden, geschaffen, unternommen oder betrieben werden muss. Modelle dienen zum einen dem Verständnis eines komplexen (Software-) Systems, wie auch zur Vermittlung von Erkenntnissen über Eigenschaften und Verhalten gegenüber anderen Interessenten an diesem Systems. In der Domäne Software Engineering bzw. Softwaretechnik wird ein Modell definiert als, „a semantically closed abstraction of a subject system“ (ISO/IEC 15474-1:2002)[2]. Anhand dieser Definition lassen sich bereits erste Schwierigkeiten ausmachen, die diese Aufgabe mit sich bringt. Anstelle von Abstraktion wird auch von Modellbildung gesprochen. „Durch das Abstrahieren vom Konkreten, erstellt man ein Modell der realen Welt, d. h. das Modell repräsentiert die reale Welt durch sein charakteristisches Verhalten“ (Balzert & Balzert, 2009, S. 27). Der Begriff des Modells wird auf den Begriff der Abstraktion zurückgeführt, der ebenso schwer zu fassen ist. Zudem bleibt offen, wer oder was in welcher Hinsicht abstrahiert.

1 Ursprünglich leitet sich der Modellbegriff vom Lateinischen bzw. Italienischen ab: modulus (lat): Maß, Regel, Form, Muster, Vorbild; "[...] modello (ital.) = Muster, Entwurf, zu lat. Modulus [...]"(z. B. www.duden.de).

2 weitere Definitionen des Modellbegriffs aus Normen und Standards in der Informatik finden sich z. B. in Software Engineering IEEE 610.12.-1990: „Model is an approximation, representation, or idealization of selected aspects of the structure, behavior, operation, or other characteristics of a real-world process, concept, or system.“ Auch in IEEE Standard 1233-1998(R2002) wird der Modellbegriff definiert: „Model is a representation of a real world process, device or concept.“ (IEEE Standard 1320.2.-1998(R2004) definiert Modell als „a representation of something that suppresses certain aspects of the model subject.“ In der ISO/IEC 15474-1:2002 wird ein Modell beschrieben als „a related collection of instances of meta-objects, representing (describing or prescribing) an information system, or parts thereof, such as a software product.“

„Modeling is the most important engineering technique; models help us to understand and analyze large and complex problems“ (Kramer, 2007, S. 41). Modellierung ist aber integraler Bestandteil des Problemlöseprozesses, wie ihn auch Pòlya beschreibt: „Draw a figure“(s.o.).

Sobald Menschen ein komplexes Problem lösen müssen oder etwas Neues, Großes erstellen möchten, machen sie sich ein Bild davon: sie erstellen ein Modell (Hesse & Mayr, 2008, S. 377). Genau diese Kompetenz benötigen auch Software Ingenieure um ein komplexes Software System zu entwickeln. Ein guter Ansatz ist es, zunächst ein Bild zu entwerfen (Polya (2010)) bzw. ein (grafisches) Modell zu erstellen (Booch und Eykholt (1996), S.80)— zu modellieren. Dies stellt jedoch eine Herausforderung für Studierende wie Lehrende dar. Einige Autoren stellen fest, dass die meisten Studienabgänger nicht in der Lage sind, ein Software- Modell zu erstellen, d. h. zu modellieren (z. B. Eckerdal, McCartney, Moström, Ratcliffe & Zander, 2007; Thomasson, Ratcliffe & Thomas, 2006a).

Die vorliegende Arbeit hat deshalb das Ziel, ein Lernangebot für Studierende zu schaffen, das auf einer wissenschaftlich untermauerten Systematik fußt und sowohl für Studierende als auch für Dozierende eine Umgebung schafft, selbstständiges Lernen, speziell auf dem Gebiet der Modellierung zu ermöglichen. Auf Basis von Problemen, die im Rahmen dieses Dissertationsvorhabens bei Studierenden während des Modellierungsprozesses mit der Unified Modeling Language (UML) als Sprache der Modellierung erhoben wurden, wurden verschiedene Scaffolds (angepasste Unterstützungsmaßnahmen) experimentell entwickelt und evaluiert. Dazu gehören klassische Hilfsmittel, wie beispielsweise die Bereitstellung von Unterlagen, Tutorials, eine Versionierung der Diagramme und verschiedene Möglichkeiten zur Kommentierung, aber auch technologiebasierte Hilfsmittel unter Verwendung der aktuellen Technologien erweiterte Realität (Augmented Reality (AR), Blickverfolgung durch Eyetracking und Techniken der Textanalyse unter Verwendung von Natural Language Processing (NLP).

Durch einen völlig neuen Zugang mit Hilfe von AR zum Lernmaterial sollen didaktische Lernhindernisse adressiert werden. Bezogen auf die Cognitive Load Theory (Abschnitt 3.2.1) gilt es den Extraneous Load und den Intrinsic Load niedrig zu halten um Schwierigkeiten mit dem Lernmaterial zu reduzieren. Zudem kann es dazu dienen, bessere Lernergebnisse zu erzielen und die Motivation beim Lernen zu erhöhen. Außerdem kann es Studierenden helfen, Lerninhalte zu verstehen (Akçayir, Akçayir, Pektaş & Ocak, 2016), indem es beispielsweise unsichtbare Konzepte, Ereignisse, und abstrakte Konzepte visualisierbar macht. Das Konzept der Blickverfolgung im Rahmen sogenannter Eye-Movement Modeling Examples soll, ebenso wie der Einsatz von AR, einen erweiterten Zugang

zum Lerngegenstand schaffen. Bei einem Eye-Movement Modeling Example (EMME) als Hilfsmittel sehen Studierende ein Video, das neben auditiver Unterstützung, auch die visuelle Unterstützung durch Blickbewegungen eines Experten enthält (Jarodzka, Balslev, Holmqvist & Nystro, 2012). NLP wird eingesetzt um, Studierenden erste Ansätze zu bieten, sich dem komplexen Sachverhalt der Modellierung zu nähern; es werden Schritte zur Analyse der Vorgaben initialisiert und erste Schritte für die objektorientierte Analyse unterstützt (siehe Balzert & Balzert, 2009). Machine Learning und Künstliche Intelligenz, beides Termini der aktuelle Forschung, wurden in der ersten Phase der Realisierung von Hilfsmitteln noch nicht betrachtet, sind aber für weitere Entwicklungen in Ergänzung zu dieser Arbeit durchaus implementierbar.

Scaffolds, wie diese, werden in eine prototypisch entwickelte Modellierungsumgebung integriert. Diese sind diagrammspezifisch und abhängig von der Art der Modellierung. Innerhalb der Software können Studierende während der Bearbeitung einer Modellierungsaufgabe diese Scaffolds nutzen, selbst neue Hilfsmittel hinzufügen und deren Nutzen individuell bewerten. Der Prototyp bietet für Dozierende außerdem die Funktionalität, Scaffolds zu konfigurieren und zu bewerten und so eine schrittweise instruktionsbasierte Unterstützung (Scaffolding) zu ermöglichen. Das System besitzt zudem die Möglichkeit, einige der identifizierten Probleme zu detektieren. Die Erkennung wird so weit geführt, dass Dozierende zu einem erkannten Problem ein Hilfsmittel empfehlen können, das der Studierende aufrufen kann, wenn er dies möchte. Über ein Reporting ist es für Dozierende möglich, die Nutzung und Bewertung der Hilfsmittel zu überwachen und diese zusammen mit den festgestellten Problemen zu analysieren und bei Bedarf zu korrigieren oder zu optimieren.

Im Zuge vorliegender Forschungsarbeit, werden die folgenden Forschungsfragen beantwortet, die als zentrale Desiderate die Arbeit leiten:

- FF1: Welche Probleme existieren für Studierende in der Modellierung?
  - FF1.1: Wie lassen sich die Probleme clustern?
- FF2: Welche Unterstützungsmöglichkeiten existieren bei der Problemlösung?
  - FF2.1: Welche Arten von Unterstützung für Software Engineering und speziell für die Modellierung gibt es?
  - FF2.2: Wie können diese Unterstützungsmöglichkeiten experimentell technologiebasiert angeboten werden?
    - FF2.2.1: Ist eine Technologie wie Augmented Reality als Unterstützungsmöglichkeit geeignet?

    - FF2.2.2: Inwiefern ist ein Eye-Movement Modeling Example als Unterstützungsmaßnahme geeignet?
  - FF2.3: Wie können diese Unterstützungsmöglichkeiten angeboten werden?
    - FF2.3.1: Wie können Unterstützungsmöglichkeiten individuell angeboten werden?
    - FF2.3.2: Inwiefern lassen sich Hilfsmittel zwischen Dozierendem und Studierendem und unter Studierenden empfehlen?

1.1 ABGRENZUNG

Es handelt sich bei diesem Forschungsvorhaben um ein interdisziplinäres Vorhaben, welches auf Basis empirischer Daten, didaktische Aspekte und Studierendeninteressen wie auch Intentionen Dozierender berücksichtigt. Unter Einbezug dieser Aspekte liegt das Hauptaugenmerk vorliegender Arbeit auf der wissenschaftlichen Fundierung sowie der prototypischen Umsetzung einer Software. Damit wird bewusst davon abgegrenzt, empirische Aussagen über eine Vergleichbarkeit von Hilfsmitteln oder Hilfsmittelarten zu treffen sowie Aussagen zur Güte des Empfehlungssystems zu tätigen. Weiterhin sind die verfügbaren Hilfsmittel und deren Angebot speziell auf die prototypische Umsetzung hin implementiert (d. h. deren erste mögliche Anwendbarkeit), sodass gerade dies Potenziale für weitere Forschungsarbeiten aufzeigt. Die Arbeit zeigt ebenfalls Ideen für den Einsatz von NLP als Mittel der Textanalyse (mehr dazu in Kapitel 9) im Kontext des Forschungsvorhabens auf. Sie liefert jedoch keine Forschungsergebnisse für die Bereiche des NLP oder künstlicher Intelligenz, sondern versteht sich vielmehr als „Türöffner“ für weitere Forschungsarbeiten speziell in diesen Bereichen.

Diese Forschungsarbeit ist eingebettet in durch das Bundesministerium für Bildung und Forschung (BMBF) geförderte Projekt Experimentelle Verbesserung des Lernens von Software Engineering (EVELIN) (Abke et al., 2012). Das Projekt EVELIN ist ein Verbundprojekt, in dem in der ersten Förderphase von 2012-2016 sechs bayerische Hochschulen, in der zweiten Förderphase von 2016-2020 fünf Hochschulen, im Rahmen des Qualitätspakts Lehre, gefördert werden. Die Hochschulen beschäftigen sich gemeinsam mit der experimentellen Verbesserung der Lehre von Software Engineering an Hochschulen.

1.2 AUFBAU DER ARBEIT

Abbildung 1.1 gibt einen Überblick über die Struktur der Arbeit. Die Struktur ist angelehnt an den Prozess des Usability Engineering (DIN Deutsches Institut für Normung e. V., 2020, S. 21) und wird im folgenden näher beschrieben.

1.2.1 *Kapitelstruktur*

Abbildung 1.1 gibt einen Überblick über die Struktur der Arbeit, aus dieser Perspektive angelehnt an den Prozess des Usability Engineering (DIN Deutsches Institut für Normung e. V., 2020, S. 21), die im folgenden näher beschrieben wird.

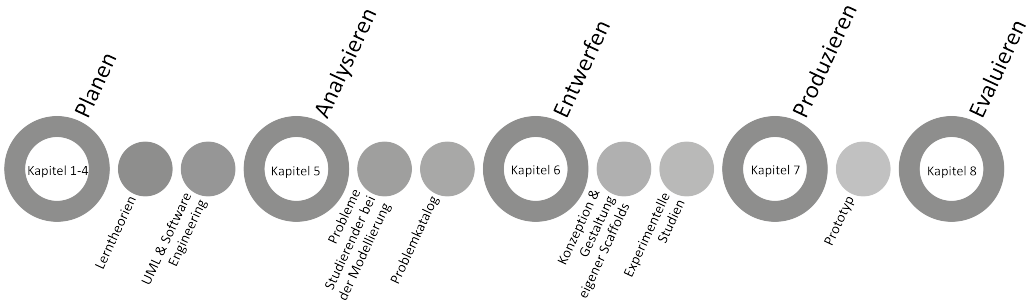


Abbildung 1.1: Struktureller Aufbau der Arbeit im Überblick

1.2.1.1 ***Kapitel 1*** *Einführung & Motivation der Arbeit:*

Das Kapitel führt in das Forschungsvorhaben ein und zeigt die Forschungsdesiderate, welche diese Arbeit behandelt und Grenzen, d. h. Themen, die diese Arbeit nicht behandelt oder lediglich anreißt.

1.2.1.2 ***Kapitel 2*** *Forschungsdesign:*

Das zweite Kapitel beschreibt das Forschungsdesign der vorliegenden Arbeit und ordnet das Vorhaben in dieses ein.

1.2.1.3 ***Kapitel 3*** *Betrachtungen zu Lerntheorien:*

Kapitel drei behandelt für die Arbeit nötiges theoretisches Hintergrundwissen aus dem Bereich der Pädagogik und Didaktik. Es werden Rückschlüsse für die Arbeit geschlossen und die beschriebenen Theorien den Bestandteilen der Arbeit zugeordnet. Während der Arbeit finden immer wieder Rückbezüge auf dieses Kapitel statt.

1.2.1.4 ***Kapitel 4*** *Die UML im Software Engineering:*

Das vierte Kapitel verortet die UML in der Disziplin Software Engineering und liefert einen Überblick über Einsatzmöglichkeiten und Verwendungszweck der UML. Es begründet den Einsatz der UML in der Software Engineering-Lehre und führt außerdem Techniken ein, die in der Objektorientierten Analyse mit der UML verwendet werden. Auch dieses Kapitel dient der theoretischen Aufarbeitung und beschreibt notwendiges Wissen aus der Disziplin Software Engineering.

1.2.1.5 ***Kapitel 5*** *Probleme Studierender bei der Modellierung mit der UML:*

Das fünfte Kapitel zeigt eine Analyse der Probleme Studierender bei der Modellierung mit der UML. Ziel der Studie war es, den Problemkontext sowie existierende Probleme bzw. Schwierigkeiten Studierender zu erfassen, zu katalogisieren und daraus Anforderungen für eine computergestützte Lernunterstützung abzuleiten. Hierfür wurde ein Mixed-Methods-Ansatz gewählt. Zunächst wurde die existierende Literatur mittels einer systematischen Literaturrecherche exzerpiert. Dabei hat sich gezeigt, dass bisher ausschließlich Artefakte Studierender analysiert wurden. Anschließend wurden mittels einer Beobachtungsstudie die Probleme Studierender während der Modellierung erfasst. Auf Basis der Probleme wurden notwendige Interventionsmaßnahmen und Berücksichtigung technologiebasierten Lernens abgeleitet. Damit wird die erste Forschungsfrage dieser Arbeit beantwortet und durch die Ableitung der Maßnahmen eine Übersicht für die Konzeption von Unterstützungsmaßnahmen geschaffen.

1.2.1.6 ***Kapitel 6*** *Konzeption eigener Scaffolds:*

Das sechste Kapitel befasst sich mit der konzeptionellen Erstellung der Scaffolds[3]. Es leitet die gewählten Unterstützungsmaßnahmen (Scaffolds) aus den vorherigen Analysen ab und zeigt die Umsetzung klassischer Hilfsmittel aber auch technologiebasierter Hilfsmittel. Hier wird auch immer wieder zurückgegriffen auf die in Kapitel drei und vier vorgestellten Theorien, um diese pädagogisch-didaktisch zu fundieren und aus Sicht des Software Engineering zu platzieren. In diesem Kapitel wird zudem die zweite Forschungsfrage beantwortet.

3 Die Begriff des Scaffolds ist bewusst nicht ins Deutsche übersetzt worden, da keine gute Abgrenzung zu Unterstützungsmöglichkeiten, Unterstützungsmaßnahmen, Hilfsmitteln und Hilfestellungen getroffen werden kann. In diesem Kapitel werden weitere Informationen dazu gegeben.

1.2.1.7 ***Kapitel 7*** *Architektur & Implementierungskonzepte der Modellierungsumgebung:*

Im siebten Kapitel liegt das Hauptaugenmerk auf der Entwicklung der Modellierungsumgebung Ariadne 2D. Es zeigt zunächst Anforderungen abgeleitet aus den vorherigen Kapiteln und beschreibt daraufhin die Systemarchitektur sowie das zugehörige Datenbankmodell als zentrale Komponenten der Software. Anschließend werden die weiteren relevanten Komponenten wie beispielsweise der schematische Aufbau der einzelnen Editoren und der Ansatz zur Detektion der Probleme beschrieben.

1.2.1.8 ***Kapitel 8*** *Evaluation:*

Das achte Kapitel beschreibt qualitativ angelegte Studien, mit dem Ziel, das allgemeine Konzept von Ariadne aus verschiedenen Perspektiven, durch Studierende und Experten zu beurteilen. In jeder der Studien steht eine Komponente von Ariadne im Vordergrund, sodass schlussendlich alle Komponenten von Ariadne evaluiert wurden: Studierende wurden zu den Editoren, zum Anlegen von Artefakten und zu Hilfsmitteln befragt. Experten wurden zur Professionalität des Werkzeugs befragt. Im Vordergrund stand bei allen Evaluationen die Usability der Umgebung.

1.2.1.9 ***Kapitel 9*** *Zusammenfassung & Ausblick:*

Das letzte Kapitel fasst diese Arbeit kapitelweise zusammen und stellt jeweils wesentliche Beiträge heraus. Darüber hinaus wird ein Ausblick zu möglichen weiteren Arbeiten gegeben, da diese Arbeit, wie einleitend beschrieben, als „Türöffner“, für weitere Forschung auf verschiedenen Gebieten dienen kann.

1.2.2 *Leserhinweise*

Dieser Abschnitt gibt Hinweise, die dem Leser mögliche Eigenheiten der Arbeit vorweg nehmen können.

- Gängige Literatur zu UML in deutscher Sprache verwendet Termini wie Requirements-Engineering oder Use-Case-Diagramm, obwohl es einen deutschen Terminus, z. B. Anforderungsmanagement und Anwendungsfalldiagramm, gibt. Um die Lesbarkeit nicht zu beeinträchtigen, werden ähnlich zu gängiger Literatur ebenfalls diese Termini, d. h. der eigentlich englische Terminus verwendet. Von der einheitlichen Verwendung der deutschen Termini wird abgesehen.

Gleiches gilt auch für Erhebungsmethoden, wie die Think-Aloud Methode, zu Deutsch Lautes Denken oder bestehende pädagogische Theorien, wie die Cognitive Load Theory (CLT).

- Die Zeitformen in dieser Arbeit wechseln. So werden beispielsweise Evaluationen bzw. bereits durchgeführte Experimente im Präteritum vorgestellt, da dies Arbeiten sind, die bereits abgeschlossen sind. Der übrige Fließtext wird im Präsens formuliert.

2

FORSCHUNGSDESIGN: DESIGN SCIENCE RESEARCH

Find the connection between the data and the unknown.

— George Pòlya (Polya, 1985, Devising a plan)

Der vorliegenden Forschungsarbeit wird der Design Science Research (DSR) Ansatz zu Grunde gelegt. Dieser Ansatz ist auf das Lösen von Problemstellungen durch die Erstellung und Evaluation von IT-Artefakten ausgerichtet. Artefakte im Design Science Research können Konstrukte, Modelle, Methoden oder Instanziierungen sein (Hevner, Ram, March & Park, 2004, S. 78). Vereint werden dabei zwei Paradigmen:

- Das „behavioral-science“ Paradigma, das Theorien aufstellt und damit menschliches Verhalten erläutert und verifiziert (Hevner et al., 2004, S. 75).
- Das „design-science“ Paradigma, das versucht durch die (Er-)schaffung neuer/innovativer Artefakte die menschlichen Fähigkeiten zu erweitern (Hevner et al., 2004, S. 75).

Hevner et al. (2004) stellen ein konzeptuelles Framework für Design Science Research vor, das beide Paradigmen vereint und Richtlinien zur Entwicklung und Evaluation von gutem Design Science Research vorschlägt.

Das Framework (siehe Abb. 2.1) besteht aus drei Blöcken, die sich zueinander in Beziehung setzen lassen: Die Realweltumgebung, Information Science Research und die Wissensbasis.

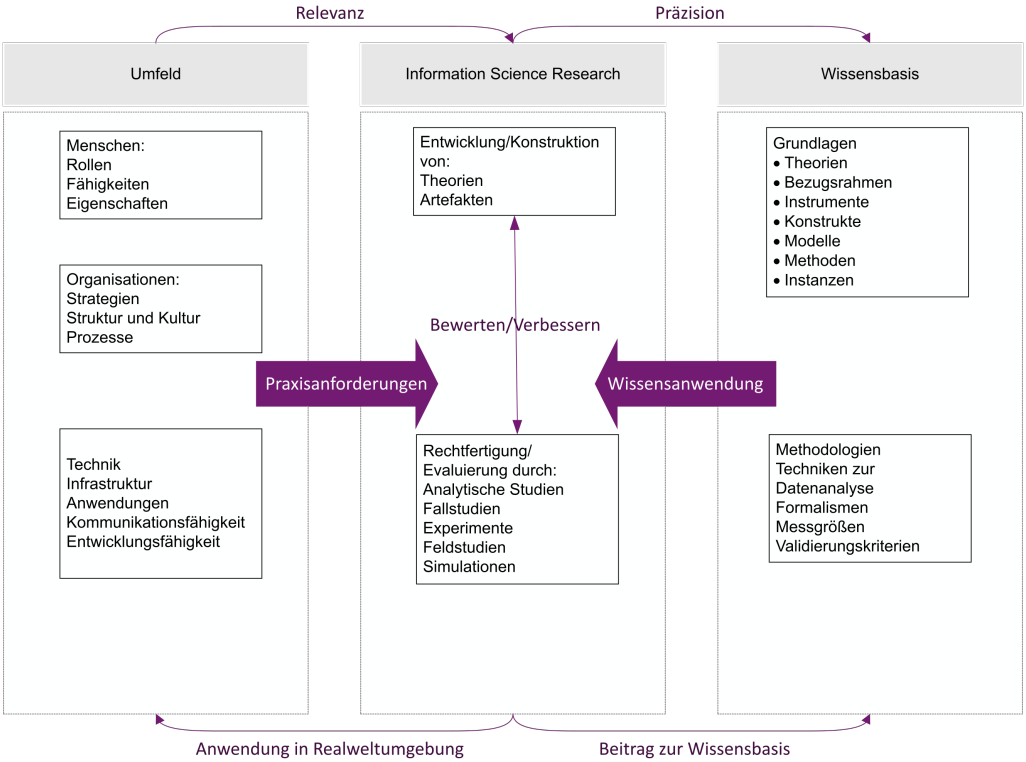


Abbildung 2.1: Framework nach Hevner ((Hevner, Ram, March & Park, 2004, S.80), eigene Übersetzung, siehe auch (Bartel, 2018, S. 6))

Hevner et al. (2004) definieren aufbauend auf Simon (1996) und Silver, Markus und Beath (1995) das Umfeld. Dieses Umfeld beinhaltet Akteurinnen und Akteure sowie geplante oder bereits existente Techniken. Ausgehend von dem Verhältnis der Akteurinnen zueinander können Ziele, Probleme, Aufgaben und Möglichkeiten abgeleitet werden. Anforderungen werden besonders durch (die Wahrnehmung der) Menschen mit ihren Rollen, Fähigkeiten und Eigenschaften beeinflusst und gegenüber aktuell existierender Technik beurteilt. Diese Anforderungen können auch als Bedarf oder Problem bezeichnet werden. Die Ausrichtung der Forschungsaktivitäten darauf gewährleistet die Relevanz der Forschung.

Sind die Anforderungen vorhanden, wird die Forschung in zwei sich ergänzenden Phasen vollzogen, die sich beide im Block des Design Science Research wiederfinden und bereits durch die Paradigmen angedeutet wurden. Es handelt sich dabei um die Entwicklung oder Konstruktion und die Rechtfertigung oder Evaluation.

Dabei entstammt der Anteil der Entwicklung und Rechtfertigung der Verhaltensforschung, um beispielsweise Theorien zu begründen, die mit den Anforderungen zusammenhängen. Ziel dabei ist Wahrheit.

Der Anteil der Erstellung und Evaluation von Artefakten, die erstellt werden, um Anforderungen gerecht zu werden, entstammt der Design Science. Ziel dabei ist Nutzen.

Beide (Wahrheit und Nutzen) sollen, laut Hevner et al. (2004), untrennbar miteinander verbunden bleiben (Hevner et al., 2004, S.79): „Truth informs design and utility informs theory“ (Hevner et al., 2004, S.80).

Wie auch im mittleren Block Information Science Research in Abb. 2.1 dargestellt, können innerhalb dessen Zyklen entstehen. Hevner (2007) nennt dies in einer späteren Betrachtung Design Cycle (Designkreislauf), da die Beurteilung durch Evaluation und Rechtfertigung dazu führen kann, dass diese Anforderungen sowohl in der Theorie als auch im Artefakt identifiziert werden und diese nochmal verfeinert bzw. neu bewertet werden müssen. In der Betrachtung von Hevner u. Chatterjee (2010) stellt das Artefakt ein Inkrement dar, das so lange iteriert, bis es die gewünschte Reife erreicht hat (siehe dazu auch Bartel, 2018, S. 6).

Die Wissensbasis bildet den dritten Block des Frameworks und besteht aus Grundlagen und Methodologien. Mit diesen kann Wissensmanagement betrieben werden, d. h. der Forscher entnimmt Trainingsmaterial (grundlegende Theorien, Rahmen, Instrumente, Konstrukte, Modelle, Methoden und Instanzen) und verwendet dieses bei der Entwicklung seiner Forschung. Methodologien kann er Richtlinien entnehmen, die die Evaluation der Artefakte stärken. Durch die An- und Verwendung von Grundlagenmaterial und Methodologien wird Präzision erzielt (Hevner et al., 2004, S.80). Dabei werden keine Vorschriften zur Verwendung der Methodologie gemacht. Der Forscher gibt der Wissensbasis seinen Beitrag zurück, wenn dieser den Anforderungen des Umfelds genügt. „A justified theory that is not useful for the environment contributes as little to the IS literature [Information Science] as an artifact that solves a nonexistent problem“ (Hevner et al., 2004, S.81).

Wie oben bereits eingeführt, stellt Hevner (2007) in einer weiterführenden Arbeit drei in das Framework eingebettete Forschungszyklen (sog. Three Cycle View) für Design Science Research Projekte vor und bringt damit eine Prozesshaftigkeit in das vorgestellte Framework. Dadurch entsteht eine Verbindung der Forschungszyklen:

- Der Relevanzzyklus (Relevance Cycle) schafft den Zusammenhang zwischen dem Umfeld/der Umgebung und den Design Science Aktivitäten.
- Der Designzyklus (Design Cycle), wie bereits oben beschrieben, zeigt die Iteration zwischen Erstellung und Evaluation der Artefakte.
- Der Präzisionszyklus (Rigor Cycle) verbindet die Säule des Design Science Research mit der Wissensbasis.

In Abb. 2.2 werden die drei Zyklen visualisiert. Zu sehen sind außerdem die Ergebnistypen des Dissertationsvorhabens für jeden Zyklus.

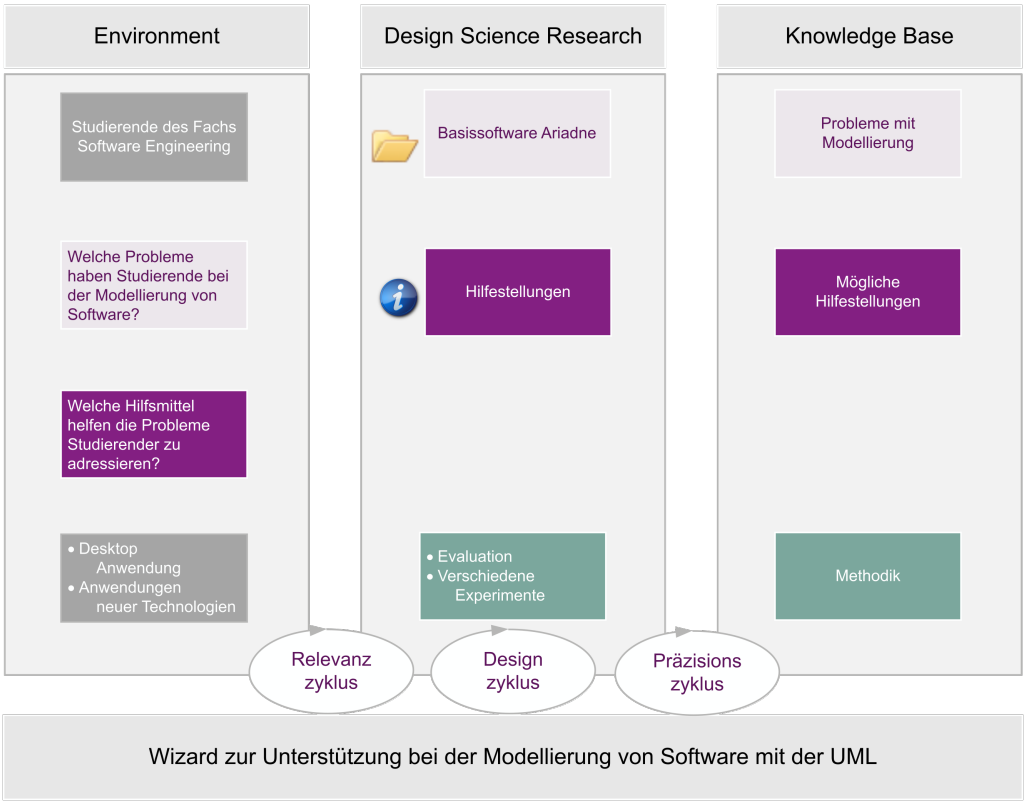


Abbildung 2.2: Three Cycle View nach Hevner angepasst an das Dissertationsvorhaben

Der Relevanzzyklus hat zum Ziel, zum einen Anforderungen für die Forschung bereitzustellen (das zu adressierende Problem), aber auch Akzeptanzkriterien zu definieren, um die Ergebnisse abschließend zu definieren (Hevner, 2007, S. 88). Es werden konkrete kognitiv erkenntnistheoretische Probleme, aber auch ressourcenbezogene und didaktische Probleme von Studierenden bei der Modellbildung mit der UML erhoben. Die Probleme werden durch Literatur, Selbst- und Fremdeinschätzung eruiert. Selbst- und Fremdeinschätzung umfassen dabei Studierenden- und Dozierendenperspektiven und werden mittels Fragebogenstudien erhoben.

Im internen Designzyklus entstehen, wie bereits oben beschrieben, Artefakte in der Konzeptionsphase, die dann in der Evaluationsphase einer gründlichen Überprüfung unterzogen werden, bevor sie im Relevanzzyklus mit dem Umfeld evaluiert werden (Hevner, 2007, S. 90).

Auf Basis der erhobenen Probleme Studierender wird im Designzyklus zunächst eine Modellierungsumgebung entwickelt. Diese wird evaluiert.

Außerdem werden experimentell verschiedene Hilfestellungen entwickelt und evaluiert. Die Hilfestellungen orientieren sich einerseits an konkreten Bedürfnissen Studierender, andererseits werden Hilfestellungen unter Verwendung aktueller Technologien (Augmented Reality, Eyetracking und Natural Language Processing) konzipiert und evaluiert. Um Studierenden Hilfsmittel bei spezifischen Problemen empfehlen zu können, ist zudem die Erfassung von Nutzerdaten notwendig. Insgesamt werden folgende Komponenten sukzessive implementiert:

1. Basissoftware zur Modellierung mit der UML
2. Hilfsmittel
3. Empfehlungssystem

Im Designzyklus entstehen Inkremente, die schlussendlich von der (Arbeits-)umgebung auf die Erfüllung der Anforderungen hin evaluiert werden. Die Methoden zur Evaluation entstammen dem Präzisionszyklus (Hevner, 2007, S. 90). Dazu werden Evaluationen mit verschiedenen Personengruppen zu den wichtigsten Komponenten der Software durchgeführt.

Der Präzisionszyklus stellt ein „Geben und Nehmen“ mit Blick auf die Wissensbasis dar. Es werden vorhandenes Wissen und geeignete bestehende Methoden aus der Wissensbasis entnommen und damit die Innovation des Projektes, aber auch eine Fundierung des Designprozesses sichergestellt. Über die Eignung hinsichtlich der Methoden entscheidet der Forscher selbst. Eine genaue Vorschrift oder ein Methodenkatalog erfolgt hier nicht. Das fertige Artefakt stellt einen Beitrag zu der vorhandenen „Knowledge Base“ dar. Dies ist laut Hevner (2007) der Schlüsselaspekt, um das Artefakt akademischem Publikum zu präsentieren.

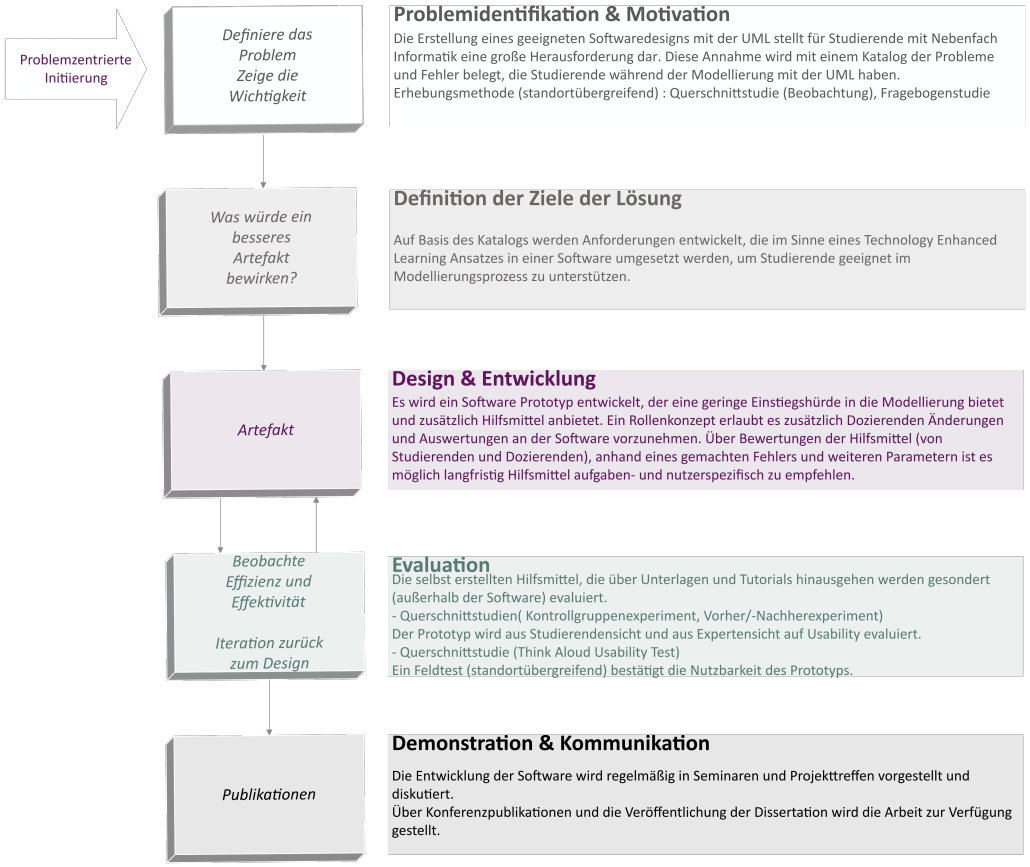


Abbildung 2.3: Design Science Research-Prozess des eigenen Vorhabens nach Peffers (Peffers, Tuunanen, Rothenberger & Chatterjee, 2007)

Dieses Dissertationsvorhaben leistet einen Beitrag zur Verbesserung der Lehre im Rahmen der objektorientierten Analyse und Design mit der UML, indem es Hilfestellungen anbietet und bei Problemen Empfehlungen dazu abgibt. Unter Verwendung von Peffers, Tuunanen, Rothenberger und Chatterjee (2007) wird der Forschungsprozess dieser Arbeit mit der „Design Science Research Methodology“ (DSRM) vervollständigt.

Peffers et al. (2007) schlagen sechs Schritte im Design Science Prozess vor: Problemidentifikation und Motivation, Definition der Ziele und Lösung, Design und Entwicklung sowie Demonstration, Evaluation und Kommunikation. In Abb. 2.3 wird der vorgeschlagene Prozess links visualisiert, daneben erfolgt eine kurze Beschreibung bezogen auf die vorliegende Arbeit.

3

BETRACHTUNGEN ZU LERNTHEORIEN: STUDIERENDENZENTRIERTES LERNEN

There are two aims which the teacher may have in view when addressing to his students a question or a suggestion of the list: First, to help the student to solve the problem at hand. Second, to develop the student's ability so that he may solve future problems by himself.

— George Pòlya (Polya, 1985, S. 26)

Entsprechend dem Design Science Research (DSR), soll bei einer problemzentrierten Initiierung zunächst die Wichtigkeit des Problems (hier: die Probleme der Studierenden bei der Modellierung eines Softwaresystems) motiviert werden. Im ersten Schritt werden zunächst Informationen aus der Wissensbasis abgeleitet und ein notwendiges Verständnis für den gewählten Ansatz zu studierendenzentrierter Lehre geschaffen. Dazu werden Theorien aus den Strömungen des Konstruktivismus und des Kognitivismus herangezogen. Der Kapitelaufbau ist deshalb wie folgt gestaltet: Es werden zunächst Theorien zur Ausgangslage vorgestellt und im Anschluss Theorien, die einen Soll-Zustand beeinflussen können, betrachtet.

Studierendenzentriertes Lernen stellt den Lerner in den Mittelpunkt, weshalb dies zunächst vorrangig erläutert wird. Lernende in den Mittelpunkt zu stellen, bedeutet auch, Probleme und Hindernisse, die das Lernen der Studierenden einschränken können, zu kennen und diese zu adressieren. Besonders wichtig ist dabei, dass Lernhindernisse individuelle Ausprägungen haben und jeder Studierende damit eine eigene Konstellation an Problemen mit sich bringt, d. h. jeder Studierende hat eine andere Ausgangslage und eine andere Lerngeschwindigkeit. Eine möglichst individuelle Adressierung der Probleme ist damit notwendig. Eine Klassifikation von Lernhindernissen wurde im Rahmen einer Vorstudie empirisch erarbeitet. Angewandte Theorien um die Klassifikation zu erarbeiten werden deshalb in Abschnitt 3.2 betrachtet. Die Abschnitte 3.3 bis 3.5 (Konstruktivistische Lernumgebungen, Technologiebasiertes

Lernen und Lehren und die Cognitive Theory of Multimedia Learning) beschreiben Theorien, die Einfluss auf die Planung des Softwaresystems hatten. Es wird angenommen, dass technologiebasierte Lehre eine effiziente und wirtschaftlich relevante Methode ist, um dies zu gewährleisten. Vor diesem Hintergrund wird in einem weiteren Abschnitt ein grundlegendes Verständnis für Technology Enhanced Learning (TEL) geschaffen.

3.1 STUDIERENDENZENTRIERTES LEHREN UND LERNEN

Betrachtet man den Terminus „Lehre“, so schwingt immer der Charakter des „Lehrens“ mit, also der Tätigkeit des Vermittelns, die ein Lehrender gegenüber eines Studierenden ausübt, während der Fokus beim Begriff des „Lernens“ klar beim Studierenden liegt. Insgesamt sind aber beide Begriffe, studierendenzentrierte Lehre (*studentcentered teaching*) und studierendenzentriertes Lernen (*studentcentered learning*), weit verbreitet. Im Zusammenhang mit diesen Begriffen werden auch die Termini lernerzentrierte Lehre (*learnercentered teaching*), Teilnehmerorientierung (*flexible learning*), erforschendes Lernen (*experiential learning*) sowie selbstorganisiertes Lernen (*self-directed learning*) verwendet (Neill & Mcmahon, 2005).

Der Grundgedanke studierendenzentrierten Lehrens und Lernens ist die Prämisse, dass Studierende ihr eigenes Lernen aktiv gestalten. Die kollektiven Arbeiten der Theoretiker wie John Dewey (1902), Jean Piaget (1950) und Lev Vygotsky (1979), die sich auf die Art und Weise konzentrierten, wie Studierende lernen, ist in erster Linie für den Übergang von klassischer Lehre, zum studierendenzentrierten Lernen verantwortlich. Ebenso werden Carl Rogers Ideen (1983) über die Bildung des Individuums prägende Arbeiten zur Entwicklung studierendenzentrierten Lernens zugeschrieben. Auch hier bedeutet studierendenzentriertes Lernen, das traditionelle lehrerzentrierte Verständnis des Lernprozesses umzukehren und die Studierenden in den Mittelpunkt des Lernprozesses zu stellen. Weiteren Einfluss auf studierendenzentriertes Lernen (hier zentrumsorientiertes Lernen genannt) hatte Maria Montessori, bei dem Vorschulkinder, durch unabhängige, selbst gesteuerte Interaktion mit zuvor vorgestellten Aktivitäten lernen.

In der Lehre werden zwei Orientierungen beschrieben: die lehrendenzentrierte/inhaltsorientierte und die studierendenzentrierte/ lernerorientierte Ausrichtung (siehe z. B. Kember (1997), Neill und Mcmahon (2005)). Letztere entspricht einer klar konstruktivistischen Haltung und beinhaltet die Sichtweise, dass Wissen von jedem Studierenden selbst konstruiert wird und der Lehrende die Rolle des „Lernermöglichers“ oder „Lernbereiters“, statt die des Informationsvermittlers einnimmt.

Harden und Crosby (2000) beispielsweise beschreiben lehrerzentrierte Lernstrategien mit dem Fokus auf dem Lehrenden, der Wissen vermittelt, wobei der Schwerpunkt auf der Vermittlung von Wissen durch Experten an Novizen liegt. Im Gegensatz dazu beschreibt Shuell (1986) das studierendenzentrierte Lernen so, dass der Schwerpunkt auf dem Lernen der Studierenden liegt und darauf, „was die Studierenden tun, um dies zu erreichen, und nicht darauf, was der Lehrer tut" (Shuell, 1986, S.429). Hannafin, Hiil und Land (1997) führen aus:

> „Conventional instructional approaches [...] reflect a positivist epistemology: information and concepts are separated from the contexts in which they naturally occur, meaning exists independent of the perceiver, and attainment of externally defined learning outcomes provides evidence of acquisition. Student centred approaches, on the other hand, are rooted in constructivist epistemology: knowledge and context are inextricably connected, meaning is uniquely determined by individuals and is experiential in nature, and the solving of authentic problems provides evidence of understanding" (Hannafin et al., 1997, S.94).

2015 wurden die Standards und Leitlinien für die Qualitätssicherung im Europäischen Hochschulraum von der European Association for Quality Assurance in Higher Education (ENQA) gemeinsam mit der European Students' Union (ESU), der European Association of Institutions in Higher Education (EURASHE) und der European University Association (EUA) im Hinblick auf studierendenzentriertes Lernen überarbeitet und damit ein Standard und folgende Leitlinien geschaffen (*Standards and Guidelines for Quality Assurance in the European Higher Education Area (ESG)*, 2015):

„Studierendenzentriertes Lernen und Lehren bedeutet in der Praxis

- die Diversität der Studierenden und ihrer Bedürfnisse zu respektieren und ihnen durch flexible Lernwege Rechnung zu tragen;
- wo es angebracht ist, unterschiedliche Vermittlungsweisen in Betracht zu ziehen und zu nutzen;
- unterschiedliche pädagogische Methoden flexibel einzusetzen;
- regelmäßige Evaluierungen und Anpassungen der Vermittlungsweisen und pädagogischen Methoden vorzusehen;
- die Studierenden zu selbstständigem Lernen zu ermutigen und ihnen als Lehrer gleichzeitig angemessene Orientierung und Unterstützung zu bieten;

- gegenseitigen Respekt in der Beziehung zwischen Lernenden und Lehrenden zu fördern;
- ein angemessenes Verfahren für den Umgang mit studentischen Beschwerden bereitzustellen."(Hochschulrektorenkonferenz, 2015, S.20; *Standards and Guidelines for Quality Assurance in the European Higher Education Area (ESG)*, 2015, S.12)

Adressiert wird studierendenzentriertes Lernen zudem im ECTS User Guide (Bologna Follow-Up Group, 2014).

Zusammenfassend lassen sich für studierendenzentriertes Lernen demnach folgende Grundsätze ableiten (siehe dazu auch Lea, Stephenson und Troy (2003); Brandes und Ginnis (1996); Neill und Mcmahon (2005)):

- unterstützt aktives statt passives Lernen,
- unterstützt konstruktivistische Epistemologie,
- betont tiefes Lernverständnis,
- weist dem/der Lernenden Verantwortung für sein/ihr Lernen zu,
- erhöht die Autonomie des Lerners,
- fordert eine Wechselbeziehung zwischen Lehrendem und Lernendem,
- fordert gegenseitigen Respekt innerhalb der Beziehung zwischen Schüler und Lehrer und eine reflexive Herangehensweise an den Lern- und Lehrprozess,
- erfolgt häufig kooperativ oder kollaborativ,
- betrachtet den Lehrenden als „Ermöglicher" und Informationsgeber,
- lässt den Lernenden in seiner Ausbildung eine Konfluenz erfahren (affektive und kognitive Bereiche fließen zusammen)(Brandes & Ginnis, 1996).

3.2 PROBLEME STUDIERENDER - LERNHINDERNISSE

Die oben definierte Umsetzung studierendenzentrierten Lernens in der Praxis ist es, „die Diversität der Studierenden und ihrer Bedürfnisse zu respektieren und ihnen durch flexible Lernwege Rechnung zu tragen" (Hochschulrektorenkonferenz, 2015, S.20; *Standards and Guidelines for Quality Assurance in the European Higher Education Area (ESG)*, 2015, S.12). Inhärent damit einher geht die Annahme, dass Studierende sehr unterschiedliche, individuelle Bedürfnisse haben, die beachtet und adressiert werden müssen. Bedürfnisse respektieren und flexible Lernwege bereiten, impliziert in dieser Arbeit auch, dass die individuellen Probleme Studierender, die sie im Lernprozess behindern, identifizier und adressiert werden. Gesprochen wird hier von einem Lernhindernis. In einer eigenen Veröffentlichung Reuter, Hauser, Gold-Veerkamp, Mottok

und Abke (2017) wurde dazu bereits eine Definition und Klassifikation vorgeschlagen, die hier in verdichteter Form dargestellt wird.

Ein Lernhindernis kann (mindestens) einer der fünf Dimensionen zugeordnet werden:

- der emotionalen Lernhindernisdimension
- der epistemologischen Lernhindernisdimension
- der didaktischen Lernhindernisdimension
- der ressourcenbezogenen Lernhindernisdimension
- der metakognitiven Lernhindernisdimension[4]

..., die einen Lernenden in irgendeiner Weise am Lernen hindern (Reuter et al., 2017, S. 10263). Ein Lernhindernis kann dabei multidimensional sein. Die (ursprünglich) sechs[4]Dimensionen von Lernhindernissen wurden aus der Cognitive Load Theory (CLT) (Sweller, 1988, 1994) (s.u.) oder der Lernstrategie-Klassifikation von Wild und Schiefele (1994)(s.u.) abgeleitet (Reuter et al., 2017, S. 10263):

1) Emotionale & motivatorische Lernhindernisse befassen sich mit motivierenden und emotionalen Aspekten, die die innere Einstellung des Lernenden abdecken.
2) Epistemologische/kognitive Lernhindernisse beschreiben eine Fehleinschätzung des Lernobjekts und/oder der individuellen, auf das Lernobjekt bezogenen, Kompetenzen.
3) Didaktische Lernhindernisse beschreiben externe Interferenzen hinsichtlich Struktur, Setting und Art des Materials für einen Kurs.
4) Ressourcenbezogene Lernhindernisse sind die komplexesten, da sie sowohl die internen Ressourcen (Aufwand und Zeitmanagement) als auch die externen (Informationsbeschaffung, Kooperation und Umwelt) betreffen.
5) Metakognitive Lernhindernisse befassen sich mit der Selbstkontrolle.
6) Psychogenetische Lernhindernisse beinhalten genetisch bedingte Dispositionen und reduzieren somit den Germane Load[4].

Die Lernhindernisklassen wurden mit Hilfe eines Fragebogens (Learning Obstacle Questionnaire) validiert und konnten mittels einer Faktorenanalyse nachgewiesen werden (Reuter et al., 2018).

4 Die Kategorie der psychogenetischen Lernhindernisse wurde ausgeschlossen, da diese nicht im Fokus der Arbeit stehen.

3.2.1 *Cognitive Load Theory*

Die CLT ist eine anerkannte Theorie und Gegenstand der Forschung im Bereich der Pädagogik (siehe z. B. Ayres & Paas, 2012; Brünken, Plass & Leutner, 2003; Figl, Mendling, Strembeck & Recker, 2010; Paas, Tuovinen, Tabbers & Van Gerven, 2003; Sweller, 1989; Sweller, van Merriënboer & Paas, 2019; Wang, Yang, Liu, Cao & Ma, 2014). Sie wurde ursprünglich von Sweller und Chandler (Sweller, 1988, 1994) aufgestellt als eine Theorie, die dem menschlichen Gedächtnis, genauer Arbeitsgedächtnis, drei verschiedene kognitive Belastungen, sogenannte Loads unterstellt:

- Intrinsic Load
- Extraneous Load
- Germane Load

Um einen erfolgreichen Lernprozess zu gewährleisten, sollten diese niedrig gehalten werden. Sweller (1988) zeigt auf, dass ein Ungleichgewicht zu einer kognitiven Überlastung (engl. cognitive overload), und damit zu Lernschwierigkeiten[5] in einem komplexen Lernprozess führen kann. Deshalb wurde die Theorie (neben der Theorie zu Lernstrategien) für die Identifikation einer Lernhindernisklassifikation ausgewählt. Die Berücksichtigung der Annahmen der CLT können zudem als Leitfaden für die Unterrichtsgestaltung verwendet werden und geben darüber hinaus Einblick in den kognitiven Prozess der Schemaentwicklung und die Übertragung von gelernten Verfahren von der kontrollierten bis zur automatischen Verarbeitung. Beide werden als Lernmechanismus bezeichnet und sind laut Sweller (1994) essentiell, um ein Thema zu beherrschen. Als Schema wird dabei die kognitive Abkürzung bezeichnet, die es dem Lernenden ermöglicht, Wissen oder ein bestimmtes Verhalten sehr schnell und effizient zu aktivieren. Bei der Erstellung von kognitiven Schemata sind alle drei Loads (also Formen der kognitiven Belastung eines Lerners) der CLT beteiligt (Reuter et al., 2017, S. 10263):

- Intrinsic (Cognitive) Load: Diese Belastung steht in direktem Zusammenhang mit den Lerninhalten. Je komplexer der Inhalt, desto höher ist die kognitive Belastung. Für die Lernenden ist es schwieriger, eine abstrakte Information (z. B. eine Programmiersprache) zu verstehen, als etwas zu lernen, das ihnen vertraut ist. Ein Lehrender kann diese Belastung nicht direkt kontrollieren (Reuter et al., 2017; Sweller, 1988).

5 Sweller nennt den Begriff der Lernschwierigkeit im selben Kontext mit „problem solving difficulty“ also übersetzt Schwierigkeit zur Problemlösung.

- Extraneous (Cognitive) Load: Die extrinsische kognitive Belastung wird durch eine ungeeignete Lernumgebung verursacht. Dies geschieht oft, wenn die erforderlichen Informationen den Lernenden in einer unangemessenen Weise präsentiert werden. Im Gegensatz zur intrinsischen Belastung kann der Lehrende diese Belastung direkt kontrollieren und durch die Verwendung eines geeigneten didaktischen Konzepts und gut gestalteter Lehrmaterialien minimieren (Reuter et al., 2017; Sweller, 1988).
- Germane (Cognitive) Load: Im Gegensatz zur intrinsischen und extrinsischen Belastung ist die lernbezogene Belastung für den Lernprozess verantwortlich. Dieser kann auch vom Lehrenden beeinflusst werden. Dieser sollte versuchen, sicherzustellen, dass der Lernprozess die Lernenden fördert und nicht als Belastung für sie erscheint.

Die drei Arten der kognitiven Belastung (Loads) addieren[6] sich, wobei der Extraneous Cognitive Load grundsätzlich niedrig gehalten werden sollte. Der Germane Load und der Extraneous Load dürfen beide hoch sein, wenn der Intrinsic Load zu diesem Zeitpunkt niedrig ist. Wichtig ist, dass die gesamte Belastung des Arbeitsgedächtnisses verhältnismäßig niedrig gehalten wird. Als Grundprinzip für die Unterrichtsgestaltung erwähnt Sweller (Sweller, 1988, 1994), dass jede Lernumgebung so gestaltet werden sollte, dass der Extraneous Load minimiert wird, und der Intrinsic Load maximiert wird. Dabei sollten auch geeignete didaktische Methoden für die Lerninhalte gewählt werden. Dies ermöglicht eine teilweise Beeinflussung der intrinsischen kognitiven Belastung (Reuter et al., 2017). Das Lernen von komplexem Material, wie es diese Arbeit dem Lernmaterial zu Software Engineering unterstellt, führt also zu einem hohen Intrinsic Cognitive Load, kann aber bei schlechtem Unterrichtsdesign zu einem gleichzeitigen hohen Extraneous Load und damit zu einer Überlastung des Arbeitsgedächtnisses der Studierenden führen.

Zu den Möglichkeiten, die gefunden wurden, um im Unterricht Extraneous Cognitive Load zu vermeiden und die Germane Cognitive Load zu optimieren, gehören beispielsweise sogenannte „worked examples“ (Chandler, Kalyuga, Tuovinen & Sweller, 2001), die als „Vorgänger“ zum sogenannten Scaffolding, wie es später in dieser Arbeit noch eingeführt wird, entwickelt wurden. Laut Sweller (2003) ist das Ziel eines Lernprozesses die Förderung der Schemaentwicklung bzw. Schemaverwendung bei Studierenden. Durch die Anwendung der Schemata wird die Belastung des Arbeitsgedächtnisses reduziert und schließlich mehr Information verarbeitet (Sweller, 2003). Ein Schema wird dabei verstanden als,

6 Addition bedeutet in diesem Fall, dass die Höhe der einzelnen kognitiven Belastung zu einer Gesamtbelastung beiträgt.

„domain-specific knowledge structures, held in long-term memory that allow people to categorize various problem solving states and determine the most appropriate moves“ (Chandler et al., 2001, S. 579).

Von dieser Theorie wurden drei Hindernisklassen abgeleitet:

- Die Klasse der psychogenetischen Hindernisse wurde aus dem Germane Load abgeleitet, da sich die Höhe dieses Loads daraus ergibt, welchen Aufwand ein Lernender betreiben muss, um das zu lernende Material zu verstehen und sein Schema zu entwickeln. Ein Lernender mit Dispositionen kann also möglicherweise die notwendige Höhe des Loads nicht erzielen.
- Die Klasse der didaktischen Lernhindernisse wurde aus dem Extraneous Cognitive Load abgeleitet. Damit sind auch Belastungen gemeint, die vom Lernen abhalten, z. B. eine ungeeignete Lernumgebung oder ungeeignetes Lernmaterial.
- Die Klasse der epistemologischen Lernhindernisse wurde aus dem Intrinsic Cognitive Load abgeleitet, da diese aus der Komplexität des Lernmaterials hervorgeht. Intrinsische Belastung ist nicht beeinflussbar und bezieht sich darauf, wie schwer ein Lernmaterial von einem Lernenden empfunden wird.

3.2.2 *Lernstrategien*

Lernstrategien haben, genau wie Lernhindernisse, bezogen auf den spezifischen Lerngegenstand, völlig individuelle Konstitutionen, tragen aber im Gegensatz zu den Lernhindernissen zum Lernerfolg bei (z. B. Boerner, Seeber, Keller & Beinborn, 2005; Lompscher, 2005; Wuttke, 2000). In Grüner (2011) wird beispielsweise der Zusammenhang zwischen Lernstrategien und Prüfungsangst bestätigt. Aus dieser Arbeit ist die Idee entstanden, von Lernstrategien (in negativer Weise, z. B. dadurch, dass sie nicht verfügbar sind) auf Lernhindernisse zu extrapolieren. Der aktuelle Stand der Literatur liefert zwar keinen Input bezüglich einer Klassifikation von Lernhindernissen, aber einige Ansätze für Lernstrategieklassifikationen (Friedrich & Mandl, 1992; Weinstein & Mayer, 1983) und Fragebögen (Pintrich, Smith, Garcia & Mckeachie, 1991; Schmeck, Geisler-Brenstein & Cercy, 1991; Weinstein, Palmer & Schule, 1987) zu diesem Thema (Reuter et al., 2018, S.458).

Im Rahmen der Erstellung der Lernhindernisklassifikation wurden deshalb auch Lernstrategieklassifikationen betrachtet und Lernhindernisklassen daraus abgeleitet. Die folgende Ausarbeitung findet sich in ähnlicher Weise in den Ausarbeitungen von Reuter et al. (2017) und

Reuter et al. (2018)[7], die der Verschriftlichung der vorliegenden Arbeit vorangegangen sind.

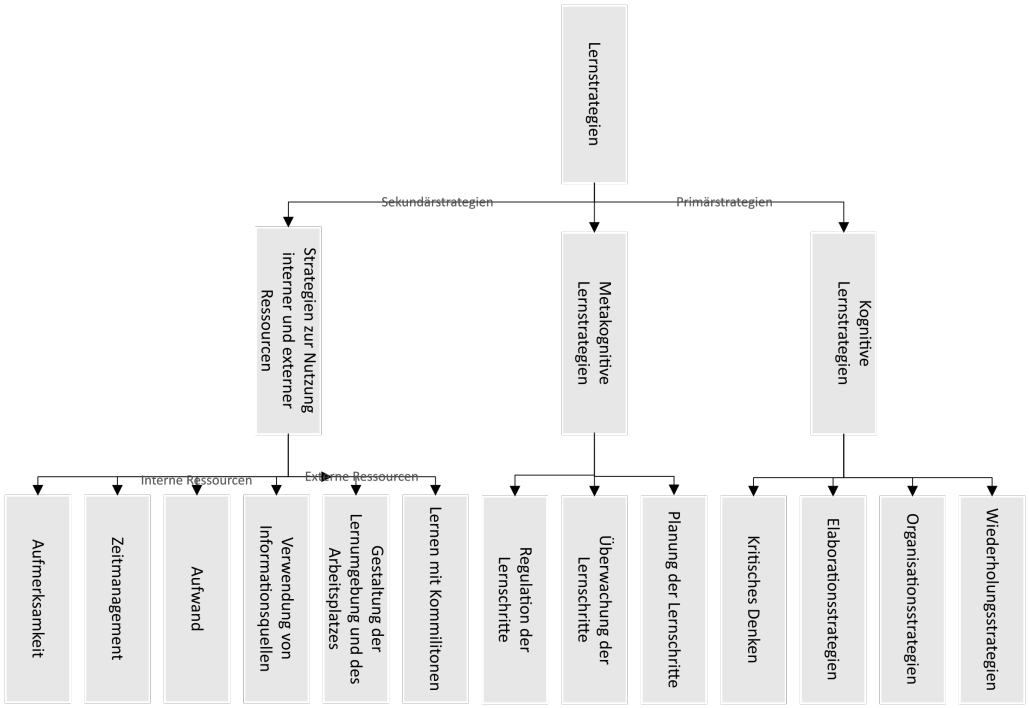


Abbildung 3.1: Klassifikation der Lernstrategien nach (Wild, 2005; Wild & Schiefele, 1994), siehe auch (Reuter, Hauser, Gold-Veerkamp, Mottok & Abke, 2017)

Der Begriff Lernstrategie ist nicht durch eine konkrete Definition beschrieben, sondern vielmehr durch ein Konzept zur „Beschreibung kognitiver und verhaltensbezogener Lernaktivitäten“ (Wild & Schiefele, 1994, S.185). Definitorisch lehnt sich die Arbeit an Ansätze von Baumert (1993), Friedrich und Mandl (1992) und Weinstein und Mayer (1983) an, die Lernstrategien als geplante Handlungsabläufe beschreiben, um ein Lehr-/Lernziel zu erreichen. Zu den Lernstrategien gehören auch Verhaltensweisen und Gedanken, die die Lernenden aktivieren, um ihre Motivation und den Prozess des Wissenserwerbs zu regulieren (Martsch & Schulz, 2015; Weinstein & Mayer, 1983). Für die Betrachtung von Lernstrategien und deren Erfassung wurde der Informationsverarbeitungsansatz gewählt, der sich an empirisch gestützten Modellen orientiert, und kognitive und motivationale Aspekte trennt (Reuter et al., 2017, S. 10261).

In Abb. 3.1 wird die Lernstrategieklassifikation von Wild und Schiefele (1994) vorgestellt, die benötigt wird, um Lernhindernisklassen abzuleiten. Sie basiert auf der Klassifikation von Weinstein und Mayer (1983) und der

7 ©2018 IEEE

Weiterentwicklung von Pintrich et al. (1991). Dieses Klassifikationsschema wurde gewählt, da es auf einem konstruktivistischen Verständnis (siehe auch Abschnitt 3.3) von Lernprozessen beruht (Martsch & Schulz, 2015; Pintrich et al., 1991; Reuter et al., 2017).

Die Klassifizierung ist zweistufig (Abb. 3.1). Die erste Stufe beschreibt drei Kategorien: Kognitive Lernstrategien, metakognitive Lernstrategien und Strategien zum Ressourcenmanagement. Die Unterskalen stellen die tatsächlich konkret nachweisbaren Lernstrategien dar. Unter kognitiven Lernstrategien werden kritisches Denken, sowie Ausarbeitungs-, Wiederholungs- und Organisationsstrategien subsumiert. Den metakognitiven Lernstrategien werden Aspekte der Planung, Überwachung und Regulierung zugeordnet. Die letzte Kategorie beschreibt Strategien für interne und externe Ressourcen. Dies ist eine größere Kategorie und beinhaltet Aufmerksamkeit, Zeitmanagement und Aufwand, die den internen Ressourcen zugeordnet werden, da sie sich mit dem „inneren" Verhalten des Lernenden ohne externe Einflussnahme befassen, während Kooperation, die Schaffung einer Lernumgebung und die Nutzung von Informationsressourcen die externen Ressourcen des Lernenden betreffen. Neben der Klassifikation Wild und Schiefele (1994), die nur kognitive Aspekte berücksichtigten, werden zusätzlich motivationale/ emotionale Lernstrategien von Friedrich und Mandl (1992) betrachtet, da auch diese Aspekte im Lernprozess eine Rolle spielen können (Reuter et al., 2017, S. 10262).

Mit dem Motivated Strategies for Learning Questionnaire (MSLQ) von Pintrich et al. (1991), dem Learning and Study Strategies Inventory (LASSI) von Weinstein et al. (1987) und dem Inventory of Learning Processes (ILP) von Schmeck et al. (1991) wurden Fragebogeninstrumente vorgestellt, um Lernstrategien zu erfassen, wobei der MSLQ nachweislich valide und sehr präzise ist. Seine Reliabilität und Validität wurde in mehr als 50 Studien untersucht und bestätigt (Artino, 2005; Pintrich et al., 1991; Pintrich, Smith, Garcia & Mckeachie, 1993; Reuter et al., 2018, S. 458).

Der MSLQ erfasst kognitive, metakognitive und ressourcenbezogene Lernstrategien sowie Items zu motivationalen Aspekten, wie Ziele und Werte für den (untersuchten) Kurs, Einschätzungen zum Erfolg im Kurs und Prüfungsangst. Auf Basis dieser Kategorien wurden drei weitere Lernhindernisklassen abgeleitet (Reuter et al., 2017, S. 10263):

- Die Klasse der emotionalen Lernhindernisse, die motivationale und emotionale Aspekte beinhaltet. Enthalten sind extrinsische, wie fehlende Belohnung oder schlechte Noten oder auch intrinsische Faktoren, wie beispielsweise fehlende Herausforderung oder Neugierde für ein Thema.

- Die Klasse der ressourcenbezogenen Lernhindernisse bezieht sich auf interne und externe Ressourcen und betrifft beispielsweise bei der Nutzung externer Ressourcen schlecht zugängliche oder nicht vorhandene Informationen, die Lernumgebung oder das gemeinsame Lernen mit Kommilitonen. Im Hinblick auf die Nutzung von internen Ressourcen betrifft es schlechtes Zeitmanagement, fehlende Aufmerksamkeit oder zu hohen bzw. zu niedrigen Aufwand (Wild und Schiefele (1994) bezeichnen dies als Anstrengung).
- Die Klasse der metakognitiven Lernhindernisse beinhaltet Lernhindernisse, die beispielweise das eigenständige Planen und Überwachen von einzelnen Lernschritten verhindern.

3.3 KONSTRUKTIVISTISCHE LERNUMGEBUNGEN

In der bereits vorab publizierten Veröffentlichung Reuter et al. (2017) wird ebenfalls ein Konzept zu sogenannten konstruktivistischen[8] Lernumgebungen (Englisch: Constructivist Learning Environment (CLE)) vorgestellt. Dies ist nicht notwendig um Lernhindernisklassen abzuleiten, aber um eine Lernumgebung zu gestalten, die konstruktivistisch und studierendenzentriert gestaltet ist. Im Folgenden werden nun die Kernpunkte, wie sie in Reuter et al. (2017) beschrieben sind, vorgestellt.

Jonassen (1999) stellt vor, wie Dozierende CLEs gestalten können und wie diese zur Anleitung der Lernenden durch die Schaffung von eigenem Wissen genutzt werden können. Sein vorgeschlagenes Modell (siehe Abb. 3.2) stellt ein Problem, eine Frage oder ein Projekt in den Mittelpunkt mit verschiedenen interpretierenden und intellektuellen Unterstützungssystemen, die es umgeben. Diese sollten authentisch sein und genügend Raum für die Manipulation des Problems (z. B. einer Aufgabe) bieten. Das Ziel ist es, dass Lernende, das Problem interpretieren und

8 Objektivismus (Behaviorismus und Kognitivismus) und Konstruktivismus werden oft als unvereinbar angesehen. Beide Strömungen unterscheiden sich in wesentlichen Annahmen. Sie haben unterschiedliche menschliche Vorstellungen und Konzepte über Lernprozesse und menschliches Verhalten. Dennoch gibt es Überschneidungen zwischen diesen beiden Strömungen (Jonassen, 1999). Einerseits ist die Hauptannahme des Objektivismus, dass Wissen von den Lehrenden auf die Lernenden übertragen werden kann. Der Lehrer kann auch durch Technologie ersetzt werden (Jonassen, 1999). Der Objektivismus sieht den Geist des Lernenden als einen Prozessor, der die Lerninhalte analysiert, neu ordnet und strukturiert. Darüber hinaus sieht der Objektivismus den Menschen als relativ passives Wesen. Er erhält nur Informationen aus seiner Umgebung und diese wiederum wird als unveränderliche Realität betrachtet. Folglich ist das Instruktionsdesign, das auf dem Objektivismus basiert, stärker auf die Idee der Informationsbereitstellung für die Lernenden ausgerichtet und es wird kein oder nur geringes Interesse an der Verarbeitung der Lerninhalte gezeigt (Cooper, 1993; Jonassen, 1985). Auf der anderen Seite basiert der Konstruktivismus auf der Idee, dass der Mensch seine eigene Realität und sein eigenes Wissen erschafft. Wissen kann daher nicht von Lehrenden an Lernende weitergegeben werden. Ein Lehrer kann nur Lerninhalte oder Erfahrungen vermitteln und den Lernprozess erleichtern (Jonassen, 1999). Auf der Grundlage der Interpretationen der Lernenden über ihre eigenen Erfahrungen wird Wissen individuell oder sozial konstruiert (Cobb, 1994, 7; Duffy & Donald, 1996; Jonassen, 1999). Im Gegensatz zum Objektivismus sieht der Konstruktivismus den Menschen als aktives Wesen. Ihr Verstand ist ein Werkzeug, das ihnen hilft, nicht nur ihr eigenes Wissen aufzubauen. Er ist ständig dabei, seine Erfahrungen zu interpretieren und seine eigene Realität zu konstruieren (Cooper, 1993; Jonassen, 1999; Jonassen, 1985). Genau genommen, gibt es Überschneidungen zwischen Objektivismus und Konstruktivismus, die sich auch in dem von Jonassen vorgeschlagenen Konzept der konstruktivistischen Lernumgebungen widerspiegeln. Beide Strömungen haben Gemeinsamkeiten, die genutzt werden könnten, um unterschiedliche Sichtweisen auf Lernprozesse und deren ideale Gestaltung zu ermöglichen (Jonassen, 1999; Jonassen, 1985). (Reuter et al., 2017, S. 10260)

lösen oder das Projekt beenden (Jonassen, 1999, S.287; Aamodt & Plaza, 1994; Reuter et al., 2017).

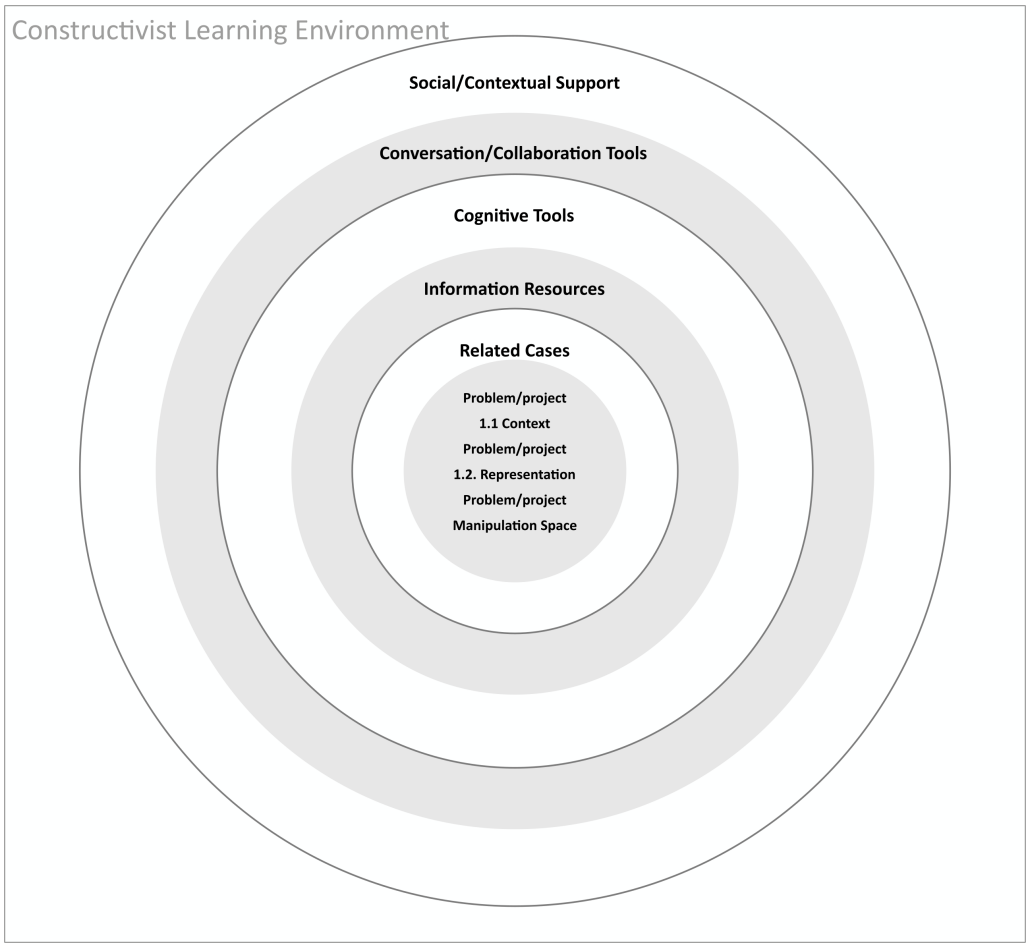


Abbildung 3.2: Konstruktivistische Lernumgebung nach Jonassen (1999)

Die folgenden Komponenten stellen Unterstützungssysteme dar:

- Verwandte Fälle (Related Cases): Diese können den Lernenden helfen zu sehen, wie andere Personen ähnliche Probleme gelöst haben. Darüber hinaus können sie einen anderen Lösungsansatz oder eine andere Sichtweise bieten und den Lernenden helfen, ihre kognitive Flexibilität zu verbessern (Aamodt & Plaza, 1994; Jonassen, 1999; Reuter et al., 2017).
- Informationsquellen (Information Resources): Um Probleme lösen und verstehen zu können, benötigt der Lerner Informationen, mit denen er sein mentales Modell (siehe Abschnitt 3.2.1) konstruieren kann. Bei der Gestaltung von CLEs, sollte der Dozierende festlegen, welche Art von Informationen der Lernende benötigt, um das

Problem zu verstehen. Vielfältige Informationsquellen sind ein wesentlicher Bestandteil von CLEs und die Lernenden sollten in der Lage sein, auszuwählen, welche Ressourcen sie wählen bzw. nutzen möchten. CLEs sollten für die Lösung eines Problems notwendige Informationen just-in-time zur Verfügung stellen. Darüber hinaus sollten sie verwendet werden, um Verwirrung oder Überlastung zu vermeiden (Jonassen, 1999; Reuter et al., 2017; Sweller, 1988, 1994).

- Kognitive Werkzeuge (cognitive Tools): Dabei handelt es sich häufig um computergestützte Werkzeuge, die den Lernenden helfen, ihr Wissen zu strukturieren und zu organisieren (z. B. Referenzmanagement, Werkzeuge zur Erstellung von Abbildungen und Tabellen usw.). Sie dienen dazu, die erforderliche kognitive Energie für den Lernprozess zu senken (Jonassen, 1999; Reuter et al., 2017; Sweller, 1988, 1994).
- Werkzeuge zur Konversation und Zusammenarbeit (Conversation/Collaboration Tools): Die Lernumgebung sollte auch die Möglichkeit bieten, mit anderen Lernenden zu interagieren. Für den Lernprozess ist es wichtig, dass das Wissen geteilt und an andere Personen weitergegeben wird. Eine CLE sollte auch kollaborative Aufgaben enthalten (Jonassen, 1999; Reuter et al., 2017).
- Soziale und kontextbezogene Unterstützung (social/contextual Support): Bei der Konzeption und Durchführung von CLEs ist die Berücksichtigung von kontextabhängigen Faktoren für eine erfolgreiche Umsetzung wichtig. Wenn Lernende schlecht ausgestaltetes Material erhalten (z. B. unprofessionelle Lernvideos), kann es sein, dass sie auch Inhalte ablehnen. Zudem ist es wichtig Lehrende zu schulen, sie weiterzubilden. Darüber hinaus sollten die Lehrkräfte in der Lage sein, einen Kurs strukturiert einzurichten und alle relevanten Materialien zu berücksichtigen (z. B. erfordert ein Software Engineering- Kurs anderes Material als eine Vorlesung in Philosophie)(Jonassen, 1999; Reuter et al., 2017).

Um das Lernen in einer CLE zu fördern, sollte ein Dozierender eher als Coach, denn als Lehrer fungieren. Eine erfolgreiche CLE benötigt Unterrichtsaktivitäten (Jonassen, 1999; Reuter et al., 2017):

- Modellierung: Der Dozierende arbeitet als Vorbild, die Leistung des Experten steht also im Vordergrund. Er oder sie zeigt den Lernenden, wie sie etwas tun können. Dieser Ansatz wird sehr oft in verschiedenen handwerklichen Fertigkeiten gesehen (Jonassen, 1999; Reuter et al., 2017). Die meisten Unterrichtseinheiten gehen davon aus, dass die Lernenden, um zu lernen, versuchen werden, wie das Modell (der Dozierende/Experte) zu agieren, zunächst durch grobe

Nachahmung, dann durch artikulierende und gewohnheitsmäßige Ausführung bis hin zu originellen Ausführungen (Jonassen, 1999, S.232).

- Coaching: Diese Aktivität zielt auf eine Verbesserung der Fähigkeiten der Lernenden ab. Die Lehrkraft beobachtet die Lernenden und gibt ihnen Feedback. Die Lernenden versuchen, selbst eine kritische Reflexion durchzuführen. Auch wenn der Lernprozess komplizierter wird, versucht der Lehrer, die Lernenden zu motivieren (Jonassen, 1999; Reuter et al., 2017).
- Scaffolding: Scaffolding ist ein eher systemischer Ansatz zur Unterstützung des Lernenden, der sich auf die Aufgabe konzentriert, die Umgebung, den Lehrer und den Lernenden. Die Lehrkraft versucht, Aufgaben zu finden, die für die Lernenden komplexer sind. Wenn sich die Fähigkeiten der Lernenden verbessern, wird der Lehrer seine Unterstützung reduzieren. Am Ende sollte der Lernende in der Lage sein, alle Aufgaben ohne Unterstützung zu lösen (Jonassen, 1999; Reuter et al., 2017). Scaffolding bietet einen temporären Rahmen, um das Lernen der Studierenden zu unterstützen, das über die kognitiven Kapazitäten der Lernenden hinausgeht. Resnick (1988) schlägt beispielweise vor, dass Aufzeichnungen und andere Hilfsmittel, insbesondere die in computergestützten Umgebungen üblichen Darstellungsmittel, als Scaffolding dienen können. Für CLEs beschreibt Jonassen (1999) Scaffolding als Maßnahme eines Systems, das durch Manipulation der Aufgabe die Fähigkeiten des Studierenden unterstützt; dies geschieht beispielweise durch Änderung der Art der Aufgabe oder den Einsatz kognitiver Hilfsmittel oder durch Änderung des Schwierigkeitsgrades. Scaffolding konzentriert sich also auf die Aufgabe an sich und nicht auf deren Ausführung. Die Anfrage eines Studierenden nach Scaffolding könnte demnach durch einen Button „Hilf mir“ in der computergestützten Umgebung repräsentiert werden (Jonassen, 1999, S.235).

Jonassen (1999) stellt fest, dass Schwierigkeiten in der Ausübung einer Aufgabe verschiedene Gründe haben können. Entweder Lernende verfügen über unzureichende Vorkenntnisse oder eine unzureichende Bereitschaft zur Ausführung der Aufgabe (Jonassen, 1999, S.235). Dies legt seiner Meinung nach drei verschiedene Ansätze für Scaffolding des Lernens nahe: die Schwierigkeit der Aufgabe an den Lernenden anpassen, die Aufgabe so umstrukturieren, dass ein Mangel an Vorkenntnissen ersetzt wird oder alternative Prüfungen (Assessments) anbieten (Jonassen, 1999, S.235).

3.4 A COGNITIVE THEORY OF MULTIMEDIA LEARNING

Die Cognitive Load Theory von Sweller und Chandler wurde bereits in den 1980er Jahren vorgestellt (siehe Abschnitt 3.2.1). Da sich unsere Gesellschaft derzeit gerade im Hinblick auf die Digitalisierung weiterentwickelt, können damit auch Veränderungen im Lernprozess einhergehen, die auch Weiterentwicklungen oder Ergänzungen bestehender Theorien erfordern, in diesem Fall um die Ergänzung des „Multimedialen Lernens".

Die Cognitive Theory of Multimedia Learning (CTML) ist relevant für den pädagogischen Ansatz zur Verbesserung der Gestaltung von multimedialen Lerninhalte (Mayer, 2014, S.46).

Die grundlegenden Elemente der Theorie sind, „dual channels", „limited capacity" und „active processing" (Mayer, 2014, S.64). Die CTML trug über die Jahre hinweg viele verschiedene Namen („model of meaningful learning" (Mayer, 1989), „cognitive conditions for effective illustrations" (Mayer & Gallini, 1990), „dual-coding model" (z. B. Mayer & Anderson, 1991), „dual- processing model of multimedia learning" (z. B. Mayer & Moreno, 1998)), zudem wurden die drei Elemente unterschiedlich fokussiert. Alle drei Elemente werden von der „generative theory" (Mayer & Anderson, 1991, S.5) und der „generative theory of multimedia learning" gleichermaßen berücksichtigt. Die häufige Namensänderung und der sich verändernde Fokus in der Theorie, lässt die Zusammenhänge zunächst kompliziert erscheinen. Abbildung 3.3 bezieht sich auf die Arbeit von Mayer (1997), da sie sowohl alle drei Elemente der Theorie beinhaltet als auch eine erste Darstellung des Modells, wie es derzeit in ähnlicher Weise immer noch verwendet wird. In der Darstellung der Theorie bezieht sich Mayer (1997) bei der Entwicklung hauptsächlich auf die generative Theorie von Wittrock (1989) und die Theorie der „dual channels" von Paivio (1990).

Aus der generativen Theorie (Wittrock, 1989) wird abgeleitet, dass sinnhaftes Lernen dann stattfindet, wenn Lernende die relevanten Informationen aus der Gesamtheit der Informationen, die ihnen zur Verfügung steht, selbst auswählen. Anschließend organisieren die Lerner die relevante Information in eine konsistente mentale Repräsentation. In einem dritten Schritt wird die neu gewonnene Repräsentation mit bereits bestehenden in Verbindung gesetzt und somit integriert (Mayer, 1997). Aus der Theorie der dualen Kodierung (Paivio, 1990) leitet er ab, dass kognitive Prozesse (genauer, die in der generativen Theorie angesprochenen) innerhalb zweier separater Informationsverarbeitungssysteme ablaufen: einem visuellen System zur Verarbeitung von visuellem Wissen und

einem verbalen System zur Verarbeitung von verbalem Wissen (Mayer, 1997).

In der generativen Theorie des multimedialen Lernens wird der Lernende als ein Wissenskonstrukteur gesehen, der aktiv Teile des visuellen und verbalen Wissens auswählt und verbindet. Die zentrale Aussage der generativen Theorie des multimedialen Lernens ist, dass die Gestaltung einer multimedialen Lernumgebung das Ausmaß beeinflusst, in dem sich die Lernenden an den kognitiven Prozessen beteiligen, die für ein sinnvolles Lernen innerhalb der visuellen und verbalen Informationsverarbeitungssysteme erforderlich sind (Mayer, 1997).

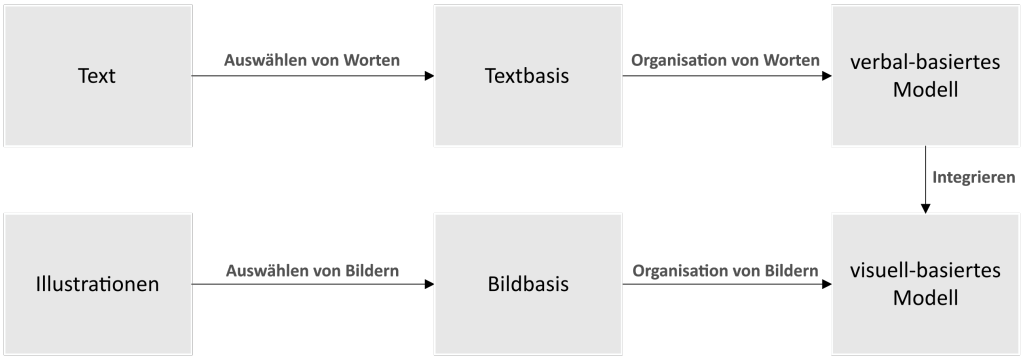


Abbildung 3.3: Cognitive Theory of Multimedia Learning

In Abb. 3.4 beschreibt Mayer (1997) ein Modell zum Lernen in einer multimedialen Lernumgebung: Sobald der Lernende visuelles und verbales Material zur Verarbeitung im visuellen bzw. verbalen Kurzzeitgedächtnis ausgewählt hat, besteht der nächste Schritt darin, das ausgewählte Material konsistent zu organisieren. Dabei spielen Aspekte der „assoziativen Verarbeitung“ von Paivio (Paivio, 1990, S. 69) eine Rolle, da Assoziationen zwischen Elementen innerhalb des visuellen Informationsverarbeitungssystems oder innerhalb des verbalen Informationsverarbeitungssystems gebildet werden. In Modellen der menschlichen Informationsverarbeitung beinhaltet die Wissensorganisation eine Transformation des verbalen Wissens innerhalb des verbalen Kurzzeitgedächtnisses und eine Transformation des visuellen Wissens innerhalb des visuellen Kurzzeitgedächtnisses (Mayer, 1997).

Der Pfeil „Organisation von Worten“ zeigt an, dass der Lernende die Textbasis in sein verbales mentales Modell reorganisiert. Diese Transformation findet im verbalen Kurzzeitgedächtnis statt (Mayer, 1997). In ähnlicher Weise zeigt der Pfeil „Bilder organisieren“ an, dass der Lernende sich seine Bildbasis neu organisiert- das visuelle mentale Modell (Mayer, 1997, S.5). Zusammenfassend lässt sich sagen, dass der Lernende innerhalb der visuellen und verbalen Informationsverarbeitungssysteme

jeweils das konstruiert, was als Situationsmodell bezeichnet wurde - eine kohärente mentale Repräsentation eines Systems, in dem die Teile auf logische Weise miteinander in Beziehung stehen (Greeno, 1989; Mayer, 1997).

Im letzten Schritt stellt der Lerner Verbindungen zwischen dem verbalen und dem visuellen Modell her. Dieser Prozess wird von Mayer (Mayer, 1984, S.33) als „Integrieren" bezeichnet und als „Verbindung der organisierten Information mit anderen bekannten Wissensstrukturen, die sich bereits im Gedächtnis befinden" beschrieben (Mayer, 1997, S.5). Dieser Prozess steht im Zusammenhang mit Paivios (Paivio, 1990, S. 69) „referenzieller Verarbeitung", bei der Verbindungen zwischen Darstellungen in den verbalen und visuellen Informationsverarbeitungssystemen bestehen (Mayer, 1997, S.5).

In Abb. 3.4 bedeutet der Pfeil „Integrieren", dass 1:1- Beziehungen zwischen den verbalen und visuellen Modellen des Materials gebildet werden. Dieser Prozess findet innerhalb des Kurzzeitgedächtnisses (oder eines Teils davon, des sogenannten Arbeitsgedächtnisses) statt. Damit der Integrationsprozess stattfinden kann, müssen die visuellen Informationen im visuellen Kurzzeitgedächtnis zur gleichen Zeit, wie die entsprechenden verbalen Informationen im verbalen Kurzzeitgedächtnis gehalten werden. Die Haltekapazität des Kurzzeitgedächtnisses ist jedoch begrenzt, so dass die integrative visuelle und verbale Information während des Lernens belastet wird. An dieser Stelle bezieht sich Mayer (1997) erstmals auf die Cognitive Load Theory (siehe v.a. Chandler und Sweller (1991)), die mit der möglichen Überlastung des Arbeitsgedächtnisses genau diese These stützt.

Zusammengefasst basiert die CTML auf drei kognitionswissenschaftlichen Prinzipien des Lernens: Das menschliche Informationsverarbeitungssystem umfasst zwei Kanäle für die visuelle/bildliche und die auditive/verbale Verarbeitung (d. h. Annahme der zwei Kanäle). Jeder Kanal hat eine begrenzte Verarbeitungskapazität (d. h. Annahme der begrenzten Kapazität). Aktives Lernen beinhaltet die Durchführung eines koordinierten Satzes von kognitiven Prozessen während des Lernens (d. h. Annahme der aktiven Verarbeitung)(Mayer, 2014). Die kognitiven Prozesse, die beim aktiven Lernen ablaufen, entsprechen den in Abb. 3.4 beschriebenen Prozessen Selektieren, Organisieren und Integrieren.

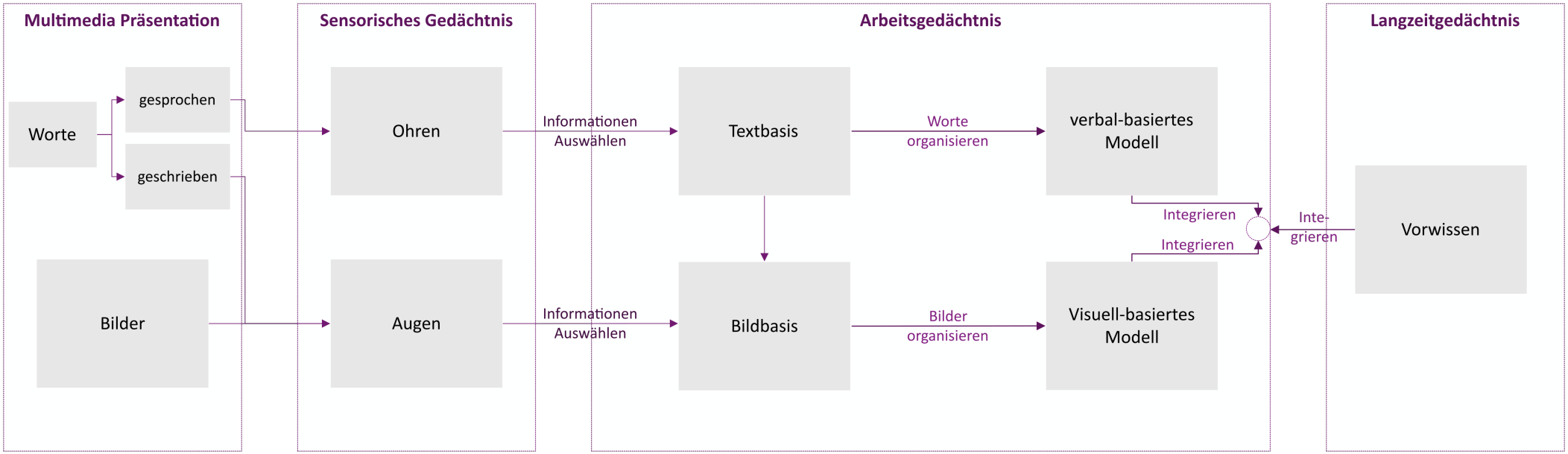


Abbildung 3.4: Generatives Modell des Multimedia Lernens von (Mayer, 1997, S.4) (eigene Übersetzung)

In Rahmen der CTML wird auch angenommen, dass das menschliche Informationsverarbeitungssystem in drei beteiligte Teile des Gedächtnisses aufgeteilt ist: das sensorische Gedächtnis, das Arbeitsgedächtnis und das Langzeitgedächtnis. Abbildung 3.3 ist ähnlich aufgebaut, wie Abb. 3.4, die in der Generative Theory of Multimedia Learning bereits gezeigt wurde. Allerdings berücksichtigt die CTML verschiedene Teile des Gedächtnisses und die Einflüsse der Außenwelt.

Durch die Präsentation von Lernmaterial gelangen Bilder und Worte in das sogenannte sensorische Gedächtnis. Das sensorische Gedächtnis ermöglicht es, Bilder und gedruckten Text als exakte visuelle Bilder für eine sehr kurze Zeitspanne in einem visuellen sensorischen Gedächtnis zu halten und gesprochene Worte und andere Töne als exakte auditive Abbilder für eine sehr kurze Zeitspanne in einem auditiven sensorischen Gedächtnis zu halten. Der Pfeil von „Bilder“ zu „Augen“ entspricht der Registrierung eines Bildes im visuellen sensorischen Gedächtnis. Der Pfeil von „Worte“ zu „Ohren“ entspricht der Registrierung von gesprochenem Text im auditiven sensorischen Gedächtnis und der Pfeil von „Worte“ zu „Augen“ entspricht der Registrierung von gedrucktem Text im visuellen sensorischen Gedächtnis (Mayer, 2014). Hinterfragt man das sensorische Gedächtnis, so würde man eigentlich noch das „Fühlen“ erwarten. Im Hinblick auf multimediales Lernen wird aber angenommen, dass dies keine Rolle spielt und vermutlich deshalb in der Theorie unerwähnt bleibt. Zusammengefasst enthält also das sensorische Gedächtnis für sehr kurze Zeit Kopien der registrierten Worte und Bilder in unbegrenzter Menge.

Das Arbeitsgedächtnis lässt die Manipulation/Neuordnung der ausgewählten Information und eine Speicherung zu. Dabei entstehen mentale Modelle in verbaler und bildbasierter Form. Es hat nur begrenzte Kapazität. Zudem ist auch der Arbeitsspeicher zeitlich begrenzt. Wenn das Integrieren von Speicher-Elementen nicht möglich ist, verblassen diese innerhalb von Sekunden. Aufgrund dieser starken Einschränkungen unseres Arbeitsgedächtnisses ist das Lernen oft schwierig, da die Aufgaben komplex sind (Stiller, 2007). Beispielsweise kann man bei der Betrachtung der Abb. 3.3 nur einige der Kästchen und Pfeile im Gedächtnis behalten. Diese bewusste Verarbeitung findet im Arbeitsgedächtnis statt. Die linke Seite des Kastens „Arbeitsgedächtnis“ stellt das Rohmaterial dar, das in das Arbeitsgedächtnis gelangt, visuelle Abbilder von Bildern und Klangbilder von Worten. Dieses wird auf den beiden Sinnesmodalitäten, von Mayer (2014) als bildhaft und verbal bezeichnet. Die rechte Seite stellt das im Arbeitsgedächtnis erstellte Wissen dar: das mentale Modell in bildlicher und verbaler Darstellung und Verbindungen zwischen ihnen (Mayer, 2014).

Der Pfeil von „Töne“ zu „Bilder“ steht für die mentale Umwandlung eines Tons (z. B. das gesprochene Wort „Katze“) in ein visuelles Bild (z. B. das Bild einer Katze). Es ist also möglich, dass verbale und bildliche Informationen die Kanalgrenzen überschreiten (Mayer, 2014). Verbale Informationen, die (leicht) imaginiert werden können oder bildliche Informationen, die (leicht) verbalisiert werden können, können den Zugang zum anderen Kanal finden (Stiller, 2007).

Im Gegensatz zum Arbeitsgedächtnis kann das Langzeitgedächtnis (rechts in Abb. 3.3) große Mengen an Wissen über lange Zeiträume hinweg speichern. Um Material aus dem Langzeitgedächtnis aktiv verwenden zu können, muss der Lerner es in das Arbeitsgedächtnis „holen“ (siehe Pfeil „Langzeitgedächtnis“ zu „Arbeitsgedächtnis“).

Erforderliche kognitive Verarbeitungsprozesse werden durch die Pfeile „Bilder auswählen“, „Worte auswählen“, „Bilder organisieren“, „Worte organisieren“ und „integrieren“ dargestellt. Diese wurden im Rahmen der generative Theory of Multimedia Learning bereits beschrieben und werden im Folgenden nun, wie von (Mayer, 2014) vorgeschlagen, tabellarisch dargestellt und kurz erläutert.

Prozess	**Beschreibung**
Worte auswählen	Der Lernende achtet auf für sie/ihn relevante Wörter in einer Multimedia-Nachricht, um eine Textbasis im Arbeitsgedächtnis zu erzeugen.
Bilder auswählen	Der Lernende achtet auf für sie/ihn relevante Bilder in einer Multimedia-Nachricht, um die Bildbasis im Arbeitsspeicher zu erstellen.
Worte organisieren	Der Lernende baut Verbindungen zwischen ausgewählten Worten auf, um ein kohärentes verbales mentales Modell im Arbeitsgedächtnis zu erstellen.
Bilder organisieren	Der Lernende baut Verbindungen zwischen ausgewählten Bildern auf, um ein kohärentes mentales Bildmodell im Arbeitsgedächtnis zu erstellen.

Prozess	Beschreibung
Integrieren	Der Lernende baut Verbindungen zwischen verbalen und bildlichen mentalen Modellen unter Einfluss seins Vorwissens auf.[a]

[a] Ziel ist dabei immer ein konsistentes Modell basierend auf den alten (Vorwissen) und neuen Informationen. „Fehler“ im mentalen Modell sind dabei nicht ausgeschlossen.

- **Worte auswählen und Bilder auswählen**
  Die beiden Prozesse Worte auswählen und Bilder auswählen unterscheiden sich nur in ihrem Kanal, weshalb sie gemeinsam beschrieben werden können. Der kognitive Prozess, der abläuft, damit eine Information aus dem sensorischen Gedächtnis in das Arbeitsgedächtnis überführt wird, ist das Selektieren, also Auswählen relevanter Information. Das bedeutet der Lerner legt Aufmerksamkeit auf Worte oder Bilder, die in einer Multimedia Nachricht, also dem Material, enthalten sind. Wenn Worte gesprochen werden, startet der Prozess im auditiven Kanal (Ohren), sonst im visuellen Kanal (Augen), wobei ein Wechsel, wie bereits beschrieben, möglich ist. Die Notwendigkeit Informationen auszuwählen, liegt an den limitierten Kapazitäten jedes Kanals, anderenfalls würde der Prozess des Auswählens nicht benötigt. Der Lerner muss also bewerten, welche Information (Text bzw. Wort oder Bilder) für die Sinnhaftigkeit der Multimediapräsentation am relevantesten ist (Mayer, 2014, S.55). Das ist aber sehr individuell und subjektiv, da jede(r) nach seiner persönlichen Motivation und Volition entscheidet, was in diesem Fall (situativ) „wichtig“ erscheint (auch mit dem Hintergrund des Vorwissens).
- **Worte organisieren und Bilder organisieren**
  Die kognitiven Prozesse der Organisation von ausgewählten Bildern oder ausgewählten Worten beinhalten die (Aus-)bildung von mentalen Modellen als verbales Modell oder bildliches Modell. Dabei bildet der Lerner Verbindungen zwischen den Stücken von verbalem Wissen oder eben Stücken von bildlichem Wissen. Der Prozess findet in dem jeweiligen Kanal statt, der auch den Auswahlprozess beeinflusst hat. Es werden nicht einfach alle möglichen Verbindungen aufgebaut, sondern eher einfache Strukturen bevorzugt. Dabei ist der Organisationsprozess nicht willkürlich, sondern eher durch das Bemühen um Sinnfindung geprägt - wie z. B. der Aufbau einer Ursache-Wirkungs-Kette (Mayer, 2014, S.56).

- **Integrieren**
  Beim Integrieren geht es darum, Verbindungen zwischen entsprechenden Teilen des bildlichen und verbalen Modells sowie mit relevantem Wissen aus dem Langzeitgedächtnis herzustellen. Dieser Prozess findet im visuellen und verbalen Arbeitsgedächtnis statt und beinhaltet die Synchronisation zwischen diesen. Der Lernende kann auch das aus dem Langzeitgedächtnis aktivierte Vorwissen nutzen, um den Integrationsprozess zu meistern (Mayer, 2014, S.57).

Ein wichtiger Aspekt in der Betrachtung der kognitiven Prozesse der CTML ist, dass diese Prozesse nicht linear auftreten müssen, der Lerner kann auf viele verschiedene Arten diese Prozesse durchlaufen. Laut Mayer (2014) wird jeder der fünf Prozesse (Auswählen und Organisieren der Worte und Bilder sowie Integrieren) beim multimedialen Lernen häufig während einer Multimedia-Präsentation auftreten. Die Prozesse werden segmentiert und nicht auf die gesamte Präsentation angewendet. Erfolgreiches multimediales Lernen erfordert, dass der Lernende diese Prozesse koordiniert und überwacht (Mayer, 2014, S. 54).

Es sollte aber dennoch das Ziel sein, den Lernprozess des Lernenden bei der kognitiven Verarbeitung des Lernmaterials zu steuern, ohne die Kapazität des Arbeitsgedächtnisses zu überlasten. Mayer (2014) stellt drei Arten von Anforderungen an das Informationsverarbeitungssystem des Lerners vor:

- **Extraneous Processing** beschreibt analog zum Extraneous Cognitive Load der CLT eine Art der kognitiven Verarbeitung, die das beabsichtigte Lehrziel nicht unterstützt und durch ein schlechtes Lehrkonzept verursacht wird. Dies ist beispielsweise dann der Fall, wenn eine Abbildung auf eine Seite gedruckt ist, die Beschreibung der Abbildung aber auf eine andere. Dadurch muss der Lerner hin und herspringen, was zu Extraneous Processing führt und kognitive Kapazitäten unnötig verbraucht. Zudem wird kein Wissen im Arbeitsgedächtnis konstruiert (Mayer, 2014, S. 59).
- **Essential Processing** bezieht sich darauf, analog zum Intrinsic Cognitive Load der CLT, präsentiertes Lernmaterial mental im Arbeitsgedächtnis zu repräsentieren. Diese Kategorie wird von der Komplexität des Materials beeinflusst. Beispielsweise ist weniger Essential Processing notwendig, um die Abbildung des Arbeitsgedächtnisses zu repräsentieren, als die des gesamten Informationsverarbeitungssystems. Aufgabe des Essential Processing ist die Auswahl relevanter Informationen aus der Präsentation und deren Organisation in der dargestellten Form (Mayer, 2014, S. 60).

- **Generative Processing** ist analog zum Germane Cognitive Load der CLT zu betrachten und wird durch die Lernmotivation des Lerners verursacht. Dabei geht es darum, dem Lernmaterial erkennbaren Sinn und Nutzen zu geben. Ein motivierender Dozierender kann Generative Processing verursachen, da dann möglicherweise mehr Aufwand in das zu verstehende Material verwendet wird. Bei Generative Processing werden die eingehenden Informationen neu organisiert und mit dem relevanten Vorwissen integriert. So führt die generative Verarbeitung zur Konstruktion eines integrierten mentalen Modells (Mayer, 2014, S. 60).

Auch hier spielen die individuellen Konstitutionen der Lernenden die größte Rolle, denn das konzipierte Lernmaterial, das bei einem Lerner genau die richtige kognitive Auslastung verursacht, verursacht bei einem anderen Lerner eine Überlastung. Genau deshalb ist es wichtig, möglichst individuelles Lernmaterial (zur individuellen Auswahl) zur Verfügung zu stellen. Um dies effektiv und effizient gewährleisten zu können, ist eine technologiebasierte multimediale Lehre unabdingbar.

3.5 TECHNOLOGIEBASIERTES LERNEN UND LEHREN

Software Engineering impliziert inhärent die Verwendung computergestützter Werkzeuge zur Entwicklung von Softwaresystemen. Dies gilt auch für die Lehre von Software Engineering. Diese Art des Lehren und Lernens kann demnach unter Technology Enhanced Learning (TEL) verortet werden. Bisher existiert keine einheitliche Definition von Technology Enhanced Learning (z. B. Bälter, 2017; Kirkwood & Price, 2014; Passey, 2019).

Deshalb werden nun einige Definitionsversuche dargestellt und TEL in seinen Eigenschaften und inhaltlicher Ausgestaltung beschrieben, sowie relevante Verknüpfungen zu vorliegender Arbeit hergestellt.

Laut Kehrwald und McCallum (2015) bezieht sich TEL auf eine Situation, in der Technologie eingesetzt wird, um die Lernerfahrung zu verbessern. Es kann sich um Formen wie Online-Learning oder Blended-Learning (eine Mischform aus Präsenz- und E-learning) handeln, genauso aber auch in anderen Lernumgebungen angewendet werden, in denen Technologie genutzt wird, um das Lehren und Lernen zu bereichern oder zu verbessern (Kehrwald & McCallum, 2015). Ähnlich dazu argumentieren Kirkwood and Price, dass der Begriff TEL mit Technologien verbunden wird, die das Lernen unterstützen sollen (Kirkwood & Price, 2014). TEL bezieht sich somit oft auf eine direkte Lernsituation im schulischen oder universitären Kontext. Auch Manouselis, Drachsler, Vuorikari,

Hummel und Koper (2011) sind prinzipiell ähnlicher Auffassung, fassen die Definition unter Einbezug von Organisationen bzw. Einrichtungen aber etwas weiter, und definieren „[...] TEL aims to design, develop and test sociotechnical innovations that will support and enhance learning practices of both individuals and organisations. It is therefore an application domain that generally covers technologies that support all forms of teaching and learning activities“ (Manouselis et al., 2011, S.1).

Bälter (2017) verfolgt einen, man könnte sagen, konstruktivistischeren Ansatz: „TEL is directed at human creation of knowledge and the human development of competence and its codification in media as heterogeneous as, e.g., courses, books, or instant messages. TEL means supporting human activity needed for knowledge creation and competence development with tools that afford isolated or collaborative endeavors in formal and informal situations“ (Bälter, 2017, S. 1). Er stellt die Generierung von Wissen, die ein Lernender als Ziel hat, in den Vordergrund, während die vorherige Literatur eher die Verwendung von Technologien fokussiert. Insgesamt führen alle Definitionen zu folgender Beschreibung: Technologiebasiertes Lernen kann sowohl formal wie auch informell stattfinden und impliziert Technologien, die Lernen unterstützen und fördern. Es werden sowohl Lerner und Organisation (i.d.R. repräsentiert durch den Lehrenden), sowie alle Formen von Lehr- und Lernaktivitäten unterstützt.

3.6 Zusammenhänge und Fazit

Ziel dieser Arbeit ist auch eine prototypische Umsetzung einer Modellierungsumgebung mit technologiebasierten Hilfsmitteln. TEL ist somit die Lehr/-Lernform, die hier zugrunde gelegt wird, um studierendenzentrierte Lehre zu gestalten und sowohl Lernende wie Lehrende (Repräsentanten der Organisation) zu unterstützen. Möglichst individuelles Lernen setzt Technology Enhanced Learning (TEL) voraus, da nur so, verhältnismäßig, der Aufwand des Dozierenden und das Ziel des Lernenden vereinbart werden kann.

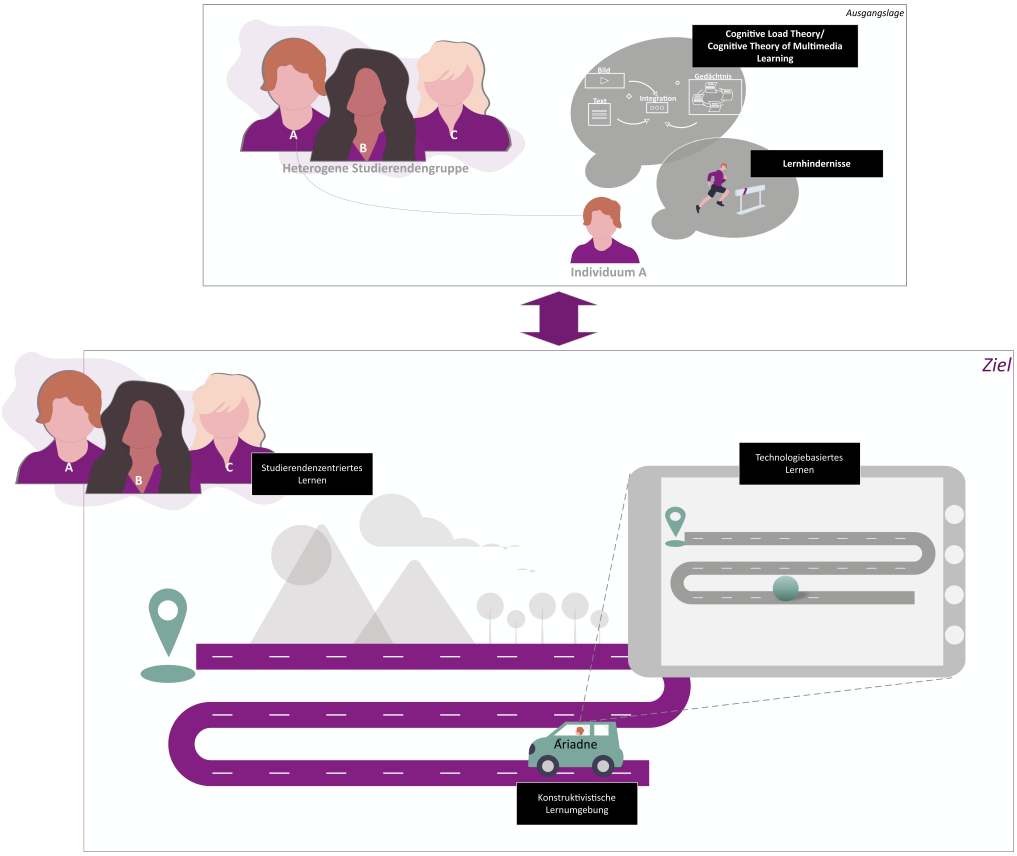


Abbildung 3.5: Zusammenhang der Lerntheorien in der Arbeit

Abb. 3.5 zeigt die metaphorisch die Zusammenhänge der einzelnen vorgestellten Theorien auf. Die Personengruppe (ABC) stellt eine (heterogene) Gruppe von Studierenden dar, die, jeder für sich, individuelle Ausprägungen von kognitiven Prozessen mit sich bringen, die durch die CLT bzw. CTML beschrieben werden können. Jede Person integriert individuell die Informationsaufnahme in den Kanälen Text und Bild im Gedächtnis. Ebenfalls individuell können Lernhindernisse existieren, die durch die Hürden symbolisiert werden. Exemplarisch ist nur eine Person (A) dargestellt. Betrachtet man die Ausgangslage spielen die Theorien zur CTML und zu Lernhindernissen (deren Klassifikation aus CLT und Lernstrategie-Theorien abgeleitet sind) eine Rolle auf individueller Ebene. Betrachtet man die Studierendengruppe, so ist es notwendig, studierendenzentrierte Lehre unter Berücksichtigung der individuellen Konstitutionen zu gestalten. Eine Zielsetzung dabei kann eine konstruktivistische Lernumgebung, beeinflusst von technologiebasierten Hilfsmittel (Kreis innerhalb), sein, die dazu beiträgt, studierendenzentrierte Lehre zu gestalten, die aber individuelle Bedürfnisse respektiert. Symbolisiert wird diese Umgebung durch das fahrende Auto, das Studierende auf dem Weg zur Lösung eines Problems in der Lernumgebung begleitet.

Über eine „Navigation“ werden Studierenden Unterstützungsmaßnahmen angeboten, die sie verwenden können (aber nicht müssen).

4

FORSCHUNGSFELD: DIE UML IM SOFTWARE ENGINEERING

The choice of what models to create has a profund influence upon how a problem ist attacked and how a solution is shaped.

— Grady Booch (Booch & Eykholt, 1996, S. 80)

Die folgenden Abschnitte verorten die Unified Modeling Language (UML) in der Disziplin Software Engineering. Dabei zeigen sie einen Abriss über Einsatzmöglichkeiten und Verwendungszweck der UML (Abschnitte 4.1 bis 4.3). Der Einsatz der UML in der Software Engineering-Lehre wird reflektiert (siehe Abschnitt 4.6) und zudem Techniken (Abschnitte 4.4 bis 4.5) eingeführt, die in der objektorientierten Analyse mit der UML verwendet werden.

Die Beschreibung bestehender, zu entwickelnder oder geplanter (Software)-Systeme ist eine zentrale Teilaufgabe im Software Engineering. Häufig wird dies mit grafischen Modellen der Unified Modeling Language (UML) durchgeführt. Es gibt auch andere Notationen wie beispielsweise Architekturbeschreibungssprachen (Architectural Description Language, ADL) vorgeschlagen von Bass, Clements und Kazman (2003), BPMN (Business Process Modeling Notation) oder eEPK (erweiterte ereignisgesteuerte Prozesskette) zur Beschreibung von Softwaresystemen (Balzert & Balzert, 2009).

„UML ist sowohl für die Modellierung von Objekten mit komplexen Beziehungen untereinander als auch für die Modellierung von Abläufen mit Nebenläufigkeits- und Echtzeitanforderungen geeignet“ (Hitz & Kappel, 2003, S.6) und wird, da sie als de facto Standard (Rupp, Queins & die SOPHISTen, 2012; Sommerville, 2013) gilt, in dieser Arbeit untersucht und diskutiert. Die Object Management Group (OMG)[9], das wichtigste Standardisierungsgremium für objektorientierte Systementwicklung (Hitz & Kappel, 2003, S.5), untertitelt die UML seit Version 1.1. als:

9 https://www.omg.org/index.htm

> „The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.“ (z. B. Booch, Jacobson & Rumbaugh, 2001, S. xix)

Insgesamt umfasst die UML nach aktueller Spezifikation (zum Erstellungszeitpunkt der Arbeit Version 2.5.1, 2017) 13 Diagrammtypen (Object Management Group, Inc. (OMG), 2017), die nach Kruchten (1995) in fünf Sichten gegliedert werden können (siehe Abb. 4.1). Ein Diagramm der UML stellt dabei eine Art „grafische Projektion“ bzw. Sicht, auf ein Modell dar. Weiterhin kann ein Modell auch, je nach Komplexitätsgrad, aus mehreren Diagrammen bestehen. Deshalb beschreibt ein Diagramm nur einen Teil des Gesamtsystems (Hitz & Kappel, 2003, S.7).

4.1 ANWENDUNGSGEBIETE DER UML IN UNTERSCHIEDLICHEN SICHTEN AUF EIN SOFTWARESYSTEM

Dieser Abschnitt beschreibt die von Kruchten (1995) vorgeschlagenen Sichten auf ein Softwaresystem. Es gibt dabei vier grundlegende Architektursichten, die alle über die Sicht der Use-Cases (Deutsch: Anwendungsfälle) verbunden sind (Abb. 4.1). All diese Sichten sind zu unterschiedlichen Zeiten hilfreich, man muss also für den Entwurf eines Softwaresystems in der Regel mehrere Sichten der Softwarearchitektur anbieten (Sommerville, 2013).

Die **Anwendungsfallsicht** beschreibt dabei die Systemfunktionalität des zu entwickelnden Systems aus Anwendersicht. In der UML wird dies mittels eines Use-Case-Diagramms realisiert (Hitz & Kappel, 2003, S.10).

Die **Entwurfssicht** zeigt die gesamte statische und dynamische Struktur des Softwaresystems. Geeignet dafür sind in der UML das Klassendiagramm (statische Struktur), Interaktionsdiagramme, wie beispielsweise Sequenzdiagramme, sowie Zustands- und Aktivitätsdiagramm (Hitz & Kappel, 2003, S.10) aus Architektur- und Entwicklersicht.

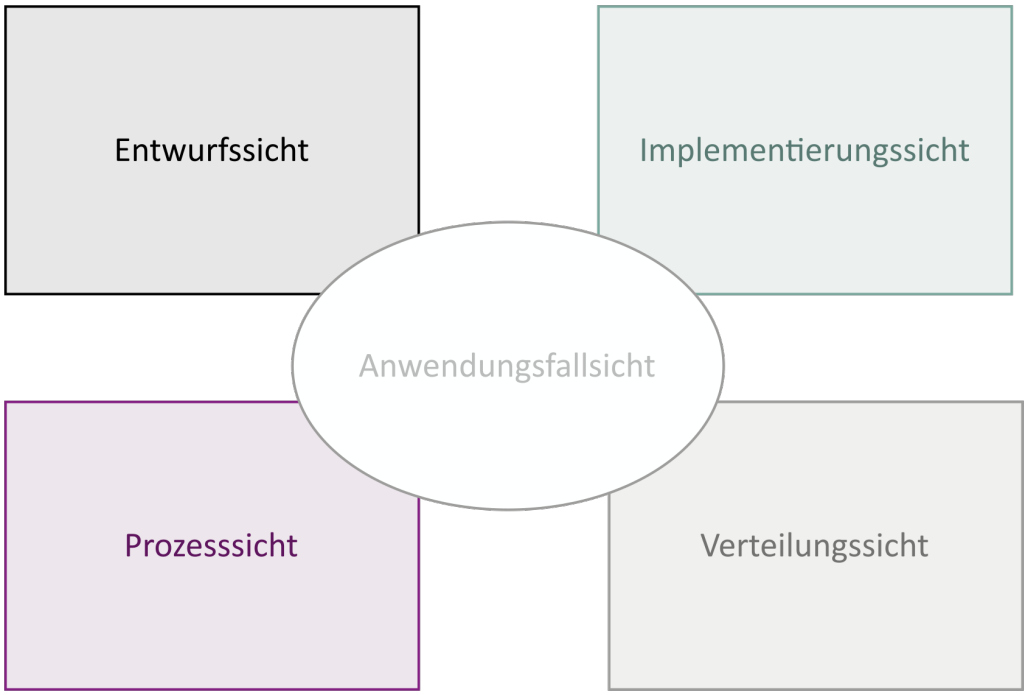


Abbildung 4.1: Sichten auf ein Softwaresystem (siehe auch (Hitz, Kappel, Kapsammer & Retschitzegger, 2005, S.9))

In der **Prozesssicht** können dieselben Diagrammtypen eingesetzt werden, allerdings mit Fokus auf Nebenläufigkeiten, der parallelen Ausführung und Synchronisation von Prozessen (Hitz & Kappel, 2003, S.10).

Die **Implementierungssicht** beschreibt die tatsächliche Aufteilung der Softwarekomponenten und deren Abhängigkeiten. Mit Hilfe der UML können diese mit einem Komponentendiagramm beschrieben werden (Hitz & Kappel, 2003, S.10).

Die **Verteilungssicht** beschreibt die Zusammenstellung der Hardware oder eine Verteilung von Softwarekomponenten und wird in der UML mit Hilfe des Verteilungsdiagramms dargestellt (Hitz & Kappel, 2003, S.10).

Während Entwurfssicht und Prozesssicht eher die konzeptionelllogische Ebene beschreiben, zeigen die Sichten Implementierungssicht und Verteilungssicht eine eher physisch-operationale Ebene (Kruchten, 1995).

4.2 Anwendungsgebiete der UML Diagrammtypen im Software Lebenszyklus

Eine weitere Möglichkeit der Gruppierung der Diagrammtypen wird in Lehrbüchern und auch Vorlesungsunterlagen (z. B. Hitz & Kappel, 2003, S.11; Kips, 2013, S. 48) durch den Einsatzzweck der Diagramme im Software Lebenszyklus Abb. 4.2 vorgegeben (siehe z. B. Frank (2000)):

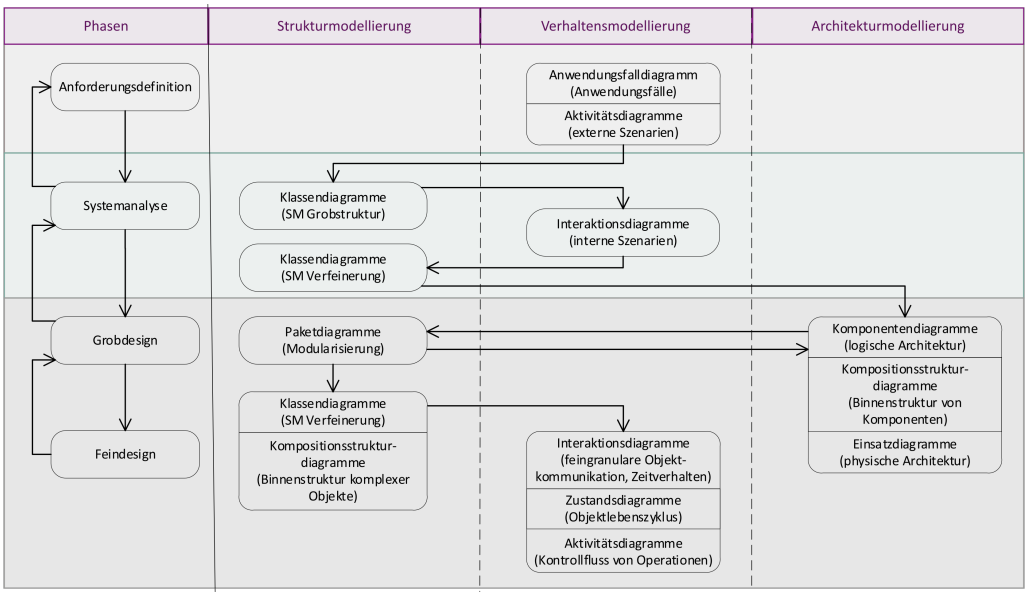


Abbildung 4.2: Zuordnung von UML Diagrammarten zum Softwareentwicklungsprozess, Abbildung adaptiert aus (Kips, 2013, S. 48; siehe auch z. B. Hitz & Kappel, 2003)

In der Phase der Anforderungsdefinition werden Anwendungsfalldiagramme und Aktivitätsdiagramme eingesetzt. In der Systemanalysephase wird sowohl Strukturmodellierung wie auch Verhaltensmodellierung durchgeführt. In der Strukturmodellierung können z. B. Klassendiagramme zum Einsatz kommen. Um Verhalten zu modellieren, können Interaktionsdiagramme modelliert werden. Nachdem Szenarien mit Hilfe von Interaktionsdiagrammen modelliert worden sind, wird das Klassendiagramm nochmals verfeinert. In der Phase des Grobdesigns wird, zunächst die logische Struktur der zu entwickelnden Software festgelegt. Deshalb kommen Komponentendiagramme zum Einsatz, für die Strukturmodellierung können beispielsweise Paketdiagramme modelliert werden. In einem Wechselspiel können im Anschluss an die Modularisierung der Systemstruktur Kompositionsstrukturdiagramme modelliert werden, um die Binnenstruktur von Komponenten zu visualisieren oder auch Einsatzdiagramme, um die physische Architektur darzustellen. In der Phase des Feindesigns werden Klassendiagramme verfeinert und weitere Interaktionsdiagramme abgeleitet, zum Beispiel, um feingranulare Objektkommunikation darzustellen oder zeitliches Verhalten. Außerdem werden Zustandsdiagramme angefertigt, um Objektlebenszyklen darzustellen und Aktivitätsdiagramme, um den Kontrollfluss von Operationen zu zeigen.

4.3 OBJEKTORIENTIERTE ANALYSE UND DESIGN MIT DER UML

Die Analyse und der Entwurf von Software Systemen nach dem objektorientierten Paradigma wird durch die Unified Modeling Language als de facto Standard (derzeit in Version 2.5) sichergestellt. Die Objektorientierte Analyse (OOA) ist „heute weitverbreitete Methode, um von den Kundenanforderungen zu einer fachlichen Lösung zu gelangen“ (Balzert & Balzert, 2009, S.548).

OOA ist ein zentrales Thema in Software Engineering Vorlesungen, die nach dem objektorientierten Paradigma lehren. Zu den für Vorlesungen zum Thema OOA eingesetzten Werken, zählen sicherlich die Werke von Helmut und Heide Balzert, wie beispielsweise das Lehrbuch der Softwaretechnik- Basiskonzepte und Requirements Engineering. Die objektorientierte Analyse werten sie als „starke Abstraktion“ (Balzert & Balzert, 2009, S.30), mit den vier verschiedene Dimensionen: Statik mit Daten und Funktionen, Dynamik und Logik (Balzert & Balzert, 2009).

„Ziel der objektorientierten Analyse ist es, die fachliche Lösung eines neuen Softwareprodukts mit Hilfe objektorientierter Konzepte zu modellieren. Die entstehende fachliche Lösung besteht aus einem statischen und einem dynamischen Modell, ergänzt um logische Elemente“ (Balzert & Balzert, 2009, S.550). Je weniger Dimensionen betrachtet werden müssen, desto einfacher fällt die Abstraktion (Balzert & Balzert, 2009, S.33). Prinzipiell „muss unterschieden werden zwischen der OOA-Methode, um von den Anforderungen zur fachlichen Lösung zu gelangen, und dem Ergebnis der Analyse, dem OOA-Modell“ (Balzert & Balzert, 2009, S.548).

Wichtig für diese Arbeit ist die OOA-Methode, die am Ende dazu führt, ein OOA-Modell zu erstellen, weil Studierende in diesem Prozess zu unterstützen. Die folgenden Ausführungen beschreiben Balzert und Balzert (2009) Vorgehen für die Erstellung eines OOA-Modells.

Es werden zunächst zwei Prozesse genannt, wobei für diese Arbeit nur der erste Prozess betrachtet werden soll:

- Der Makroprozess, der methodische Schritte vorgibt
- Die Anwendung methodischer Regeln, die in Form von Checklisten zur Verfügung stehen (Balzert, 2001; Balzert & Balzert, 2009, S.559).

"Der Makroprozess beschreibt auf einem hohen Abstraktionsniveau, in welcher Reihenfolge die einzelnen Aufgaben zur Erstellung eines OOA-Modells[10] auszuführen sind:

10 „Voraussetzung für die Erstellung eines OOA-Modells ist eine vorliegende Anforderungsspezifikation“ (Balzert & Balzert, 2009, S.559). Einen Vorschlag dafür bietet der Rational Unified Process, der verschiedene Templates zur Dokumentation von Anforderungen zur Verfügung stellt (Kruchten, 2004).

- Ermitteln der relevanten Use-Cases (oft in der Anforderungsspezifikation bereits erfolgt)
- Ableitung von Klassen aus den Use-Cases
- Erstellen des statischen Modells
- Parallel dazu Erstellen des dynamischen Modells
- Berücksichtigung der Wechselwirkung beider Modelle"(Balzert & Balzert, 2009, S.559)

Zur Ermittlung der relevanten Use-Cases gehört auch die Ermittlung von Akteuren, die die Bearbeitung eines Use-Cases auslösen und damit mit dem Softwaresystem interagieren, sich aber außerhalb von diesem befinden.

Die UML bietet dabei für jeden Schritt des Makroprozesses geeignete Diagrammtypen zur Visualisierung. In dieser Arbeit wird im Wesentlichen dieser Ansatz verfolgt, um Studierenden die Modellierung von Softwaresystemen zu vermitteln.

Für die ersten beiden Phasen werden im Folgenden nun explizit anwendbare Methoden bzw. Techniken vorgestellt, die in der Umsetzung der Arbeit im Sinne des Angebots von Hilfestellungen eine Rolle spielen.

4.4 USE-CASE TEMPLATE NACH ALISTAIR COCKBURN

Neben der Visualisierung von Use-Cases in einem Use-Case-Diagramm, bietet Alistair Cockburn eine Struktur mit der Möglichkeit an, in einem strukturierten Format Use-Cases zu erfassen (Cockburn, 2003). Mit Hilfe dieser Struktur können Use-Cases anhand definierter Kriterien komprimiert formuliert werden. Die definierten Strukturelemente (wie beispielsweise das Element Ablauf) können bei der Ableitung von Klassen helfen (siehe Abschnitt 4.5).

Tabelle 4.1: Tabellarischer Aufbau eines Use-Cases nach(Cockburn, 2003, S. 155)

Use Case #	der Titel ist das Ziel in Form eines kurzen Satzes mit aktivem Verb
Anwendungskontext	eine längere Beschreibung des Anwendungskontexts, wenn nötig
Umfang	das entstehende System als Black Box

Ebene	Auswahl: Überblick, Hauptaufgabe,Subfunktion	
Primärakteur	ein Rollenname für den Primärakteur oder seine Beschreibung	
Stakeholder und Interessen	**Stakeholder**	**Interesse**
	Name des Stakeholders	hier stehen die Interessen des Stakeholders
Vorbedingungen	der zu Grunde gelegte Stand der Dinge	
Invarianten	die in jedem Fall geschätzten Interessen	
Nachbedingungen	die beide erfolgreichem Ausgang, befriedigten Interessen	
Trigger	die Aktion im System, die den Use Case auslöst	
Beschreibung	**Schritt**	**Aktion**
	1	hier stehen die Schritte des Szenarios vom Trigger bis zur Zielausgabe und alle anschließenden Datenbereinigungen
	2	
	3	
Erweiterungen	**Schritt**	**Verzweigende Aktion**
	Ia	Bedingung, die die Verzweigung auslöst: Aktion oder Titel des Teil-Use-Case
Technik- und Datenvariationen		

1 Variantenlisten

4.5 IDENTIFIKATION VON KLASSENKANDIDATEN NACH ABBOTT TECHNIK

Für den zweiten Schritt des Makroprozesses, der Ableitung von Klassen aus Use-Cases, wird eine Methode vorgeschlagen, deren Ursprung auf Abbott (1983) zurückgeht. Seine Technik beschreibt, wie Datentypen von Gattungsnamen[11], Variablen von direkten Referenzen, Operationen von Verben und Attributen und Kontrollstrukturen von ihrer textuellen Beschreibung abgeleitet werden können. Ein Gattungsname ist der Name einer Klasse, eines Lebewesens oder Gegenstandes. Beispiele für Gattungsnamen sind „Tisch“, „Idee“, „Hase“, „Bild“ oder „Buch“. Beispiele für eine direkte Referenz sind „mein Hase“, „sein Fehler“, „der Mond“ und deuten auf ein Objekt hin. Ein Verb, Attribut, Prädikat oder ein beschreibender Ausdruck weist auf eine Operation hin (Abbott, 1983).

Abbott (1983) benennt drei Schritte, die notwendig sind, um von einer textuellen Problembeschreibung zu einem ausführbaren Programm zu gelangen:

1. Entwickeln einer informellen Strategie für das Problem
2. Formalisieren der informellen Strategie
3. Aufteilen der Lösung in zwei Teile: ein Paket und ein Unterprogramm

Die in dieser Arbeit betrachtete Technik der Substantiv-Verb-Analyse bezieht sich auf den zweiten Schritt: das Formalisieren der informellen Strategie. Hierzu sind vier Schritte durchzuführen (Abbott, 1983):

1. Identifikation der Datentypen
2. Identifikation der Objekte (Programmvariablen) dieser Typen
3. Identifikation der Operationen, die auf diese Objekte angewendet werden
4. Organisieren der Operationen in die Kontrollstruktur, die durch die informelle Strategie vorgegeben ist

Diese Handlungsanweisungen werden auch von weiteren Autoren sehr ähnlich vorgeschlagen (Seidl, Scholz, Huemer & Kappel, 2015).

11 Ein Gattungsname wird als natürlichsprachliches Äquivalent zu dem Begriff, der in Programmiersprachen für abstrakte Klassen verwendet wird, betrachtet. Ein Gattungsname benennt eine Klasse oder ein Konzept, auch wenn es dafür zum aktuellen Zeitpunkt noch keine Werte oder Operationen gibt. Gattungsnamen bieten einen konzeptionellen Blick auf ein Konzept, dass noch nicht vollständig definiert ist.

Abbott (1983) weist jedoch darauf hin, dass die vorgestellte Technik keine automatisierbare Prozedur ist. Die Identifikation der Datentypen, Objekte, Operationen und Kontrollstrukturen erfordert großes Verständnis der Realität und ein intuitives Verständnis der Problemdomäne (Abbott, 1983).

4.6 Kritische Betrachtung der UML

Derzeit ist die UML sowohl eine De-facto- (Rupp et al., 2012; Sommerville, 2013) als auch eine De-jure-Notation (ISO/IEC 19505:2012) zur Visualisierung von Modellen in der Softwareentwicklung.

Sommerville (2013) erwähnt beispielsweise drei verschiedene Absichten für die Modellierung von Systemen, die jeweils einen unterschiedlichen Grad an Detailliertheit und Strenge bei der Anwendung der Modellierungssprache erfordern (Anke & Bente, 2019; Sommerville, 2013):

- Diskussionen über ein bestehendes oder geplantes System: Diese Modelle können unvollständig sein und die Notationen der Modellierungssprache informell verwenden, beispielsweise indem nur bestimmte UML-Artefakte erstellt werden.
- Dokumentation eines bestehenden Systems: Auch diese Modelle müssen nicht vollständig sein, wenn nur Teile des Systems dokumentiert werden. Allerdings sind Korrektheit und Genauigkeit erforderlich.
- Code-Generierung: Präzise Systembeschreibung als Grundlage für die Generierung der Implementierung in einer modellbasierten Entwicklung, was bedeutet, dass ein Großteil der UML-Artefakte erstellt werden muss.

Dies entspricht im Wesentlichen auch der Systematisierung von Chaudron, Heijstek und Nugroho (2012), die zwischen Modellen zur Analyse und zum Verständnis, Modellen zur Kommunikation, Modellen als Vorlage für die Implementierung und Modellen als Grundlage für die Codegenerierung unterscheiden.

Die zentrale Rolle der Modellierung wird zudem im Software Engineering Body of Knowledge (SWEBOK) betont. Hier wird die UML als mögliche Modellierungssprache erwähnt und auf die Notwendigkeit hingewiesen, eine für die jeweilige Aufgabe geeignete grafische Notation zu wählen (Bourque, Dupuis, Abran, Moore & Tripp, 1999). Unter Anwendung der UML, werden wesentliche Richtlinien und Regeln vorgegeben, um komplexe Softwaresysteme zu visualisieren und zu verstehen. Sie definiert, als de facto Standard, Werkzeuge in Form von Diagrammen

und Diagrammabhängigkeiten. Wenn die Studierenden lernen, die Richtlinien zu befolgen und die Artefakte zu erstellen, können sie komplexe Anforderungen erstellen, die von Software-Entwicklern auch verstanden und umgesetzt werden können. Wohl auch deshalb ist die UML in vielen Lehrplänen für Software Engineering-Kurse an Universitäten weltweit (The Joint Task Force on Computing Curricula, 2001, 2004) enthalten.

Eine Sichtung der öffentlich zugänglichen Modulkataloge aller deutschen Hochschulen im Jahr 2018 mit Informatik- bzw. informatiknahmen Studiengängen ergab, dass mindestens[12] 61 von 101 Hochschulen UML in ihren Modulbeschreibungen auflisten.

Zum Vorgehen:

Auf Basis der Liste der Hochschulen in Deutschland[13] (*N=426*) wurde in einem ersten Schritt zunächst nach der Verfügbarkeit von Informatik bzw. informatiknahen Studiengängen gefiltert. In einem zweiten Schritt wurden für diese Studiengänge zugängliche Modulhandbücher recherchiert[14] und auf Software Engineering-Module (oder Ähnliche) geprüft. Aus dieser Filterung resultieren *N=101* Hochschulen mit entsprechenden infrage kommenden Studiengängen und zugänglichen Modulhandbüchern. Im Anschluss wurden die identifizierten Software Engineering bzw. Software Engineering-nahen Module der passenden Modulhandbücher nach den Begriffen „UML“ und einhergehende Termini wie „Design“, „Entwurf“ oder „Modell“ durchsucht. Eine Hochschule kann mehrere infrage kommende Studiengänge haben. Für UML wurden *N=61* Vorkommen gezählt, für Entwurf *N=71*, für Modell *N=72* und für Design *N=38*.

Die UML ist auch Teil der Empfehlungen der Deutschen Gesellschaft für Informatik (Zukunft, 2016).

Trotz ihres hohen Bekanntheits- und Verbreitungsgrades ist die UML im heutigen Software Engineering nicht unumstritten (Anke & Bente, 2019; Frank & Prasse, 1997). Ein wesentlicher Grund dafür ist laut z. B. Anke und Bente (2019) ihre unklare Anwendung in der heutigen und zukünftigen Praxis, möglicherweise bedingt durch die Differenzierung von Software-Produkttypen und agilen Entwicklungsansätzen (Anke &

12 Das Adverb mindestens soll hier aussagen, dass Modulhandbücher ständiger Änderung unterliegen und aufgrund der groben Granularität der Suchstrategien und der Sichtung möglicherweise auch nicht alle Modulhandbücher gefunden wurden. Zudem ist durch die Anzahl der nicht verfügbaren Modulhandbücher zu vermuten, dass noch deutlich höhere Ergebnisse erzielt werden würden. Die erzielten Ergebnisse sind nur als Indikator zu werten.

13 https://de.wikipedia.org/wiki/Liste_der_Hochschulen_in_Deutschland

14 Dazu wurde zunächst auf der jeweiligen Homepage der Universität recherchiert und bei erfolgloser Suche noch eine Google-Suche mit dem entsprechenden Studiengang, der Hochschule und Modulhandbuch abgesetzt.

Bente, 2019), da diese in einigen Domänen keine umfangreiche Spezifikation mehr erfordern. Vielmehr stellen konzeptionelle Modelle ein Kommunikationsmedium für die Kommunikation zwischen Software-Entwicklern, Auftraggebern und Anwendern dar (Frank, 2000). Aber trotz der Aufmerksamkeit, die derzeit Themen wie agilen Entwicklungsansätzen gewidmet wird, darf man nicht vernachlässigen, dass UML in vielen Großprojekten oder beispielsweise für sicherheitskritische Produkte immer noch eine wichtige Rolle spielt (Anke & Bente, 2019).

Anke und Bente (2019) werteten Studien über die Nutzung der UML in der Praxis aus und leiteten folgende Trends ab:

- Den ausgewerteten Studien zufolge, hat die UML einen sehr hohen Bekanntheits- und Nutzungsgrad in der Praxis.
- Die einzelnen Diagrammtypen werden mit unterschiedlicher Häufigkeit verwendet. Besonders häufig werden Klassen-, Aktivitäts-, Anwendungsfall- und Sequenzdiagramme verwendet.
- Die beabsichtigten Verwendungen befinden sich oft in der frühen Phase der Systementwicklung.
- Die Modelle sind oft informell oder mit informellen Notationen gemischt.
- Neben Papier und Whiteboard werden auch verschiedene Werkzeuge, wenn auch nicht unbedingt klassische UML-Modellierungstools, als Medien verwendet.

Wie die obigen Beschreibungen zeigen, kann die UML also im gesamten Softwareentwicklungsprozess eingesetzt werden, schreibt selbst aber keinen solchen vor. Die Diagrammtypen der UML sind flexibel im Sinne ihres Einsatzzwecks in den Phasen verwendbar und dienen als Kommunikationsmedium. Zudem überwiegen die im letzten Abschnitt beschriebenen Vorteile der UML und rechtfertigen deren Lehre und Untersuchungen zum Einsatz in der Hochschullehre.

Neben der Vorlage für die Implementierung und für die Codegenerierung, ist die UML nicht nur im industriellen Kontext ein geeignetes Mittel zur Kommunikation und zur Verständnisvermittlung, auch im Lehrkontext ist sie in der Kommunikation zwischen Dozierenden und Studierenden unumgänglich, wenn es beispielsweise um die Vermittlung von Entwurfsmustern geht. Die oben stehenden Ausführungen haben demnach aufgezeigt, dass UML trotz ihrer Komplexität ein Element im Lehrplan für Software Engineering sein sollte. Deshalb ist die UML Grundlage der vorliegenden Arbeit. Dieser Abschnitt findet sich in analoger Darstellung in Reuter et al. (2020).

5

PROBLEMZENTRIERTE INITIIERUNG: ERFASSUNG DER PROBLEME STUDIERENDER BEI DER UML-MODELLIERUNG

We build models of complex systems because we cannot comprehend any such system in its entirety.

— Grady Booch (Booch & Eykholt, 1996, S. 80)

Das fünfte Kapitel zeigt eine Analyse der Probleme Studierender bei der Modellierung mit der Unified Modeling Language (UML).

Im Sinne des Design Science Research (DSR) wird zunächst die Komplexität und Verständlichkeit der UML (Unified Modeling Language) im Studium untersucht, um die Schwierigkeiten, die sich beim Erlernen der UML bei den Studierenden ergeben, zu zeigen. Die Erhebung entspricht einer analytischen Tätigkeit, die im Sinne des DSR sowohl durch eine theoretische als auch praktische Sichtweise abgedeckt wird (Hevner & Chatterjee, 2010, S.12, S.81; Bartel, 2018, S.101). Zunächst werden Elemente der Wissensbasis (Kapitel 2) herangezogen (Rigor Cycle) und über eine systematische Literaturrecherche ((SLR), Abschnitt 5.2) bereits existierende Lernhindernisse (siehe Abschnitt 3.2), die in Form von Problemen bei der Modellierung mit der UML sichtbar werden, erfasst (Relevanzzyklus, Abschnitt 5.3.5).

Eine Erkenntnis der SLR ist die Tatsache, dass (lediglich) Artefakte Studierender als Ergebnis der Modellierung untersucht wurden. Dadurch kommt die Frage auf, ob und inwiefern sich Probleme und/oder Fehler[15], die in Artefakten gefunden wurden, von Problemen, die im Modellierungsprozess auftreten, unterscheiden. Deshalb werden mit Hilfe einer

15 Der Begriff des „Fehlers“ ist, wie auch in späteren Ausführungen deutlich werden wird, nur dann als solcher, im Sinne der UML, zu betrachten, wenn gegen das UML-Metamodell, das die UML spezifiziert, verstoßen wird. Alle weiteren Probleme beziehen sich auf falsches Verständnis oder fehlerhafte Modellierung der Logik. Richtlinien und Best Practices für die Modellierung und die objektorientierte Analyse (z. B. die Benennung von Use-Cases, Methoden im Klassendiagramm) können ebenfalls zu Problemen führen.

eigenen Erhebung zusätzliche empirische Nachweise zu Problemen Studierender während des Modellierungsprozesses erbracht. Bei der Analyse hat sich gezeigt, dass durch die Betrachtung des Modellierungsprozesses weitere Probleme Studierender aufgezeigt werden konnten, die durch die ausschließliche Analyse der Artefakte der Studierenden nicht ersichtlich wurden.

Auf dieser Basis werden anschließend notwendige Interventionsmaßnahmen unter Berücksichtigung technologiebasierten Lernens abgeleitet (siehe Abschnitt 5.3.6). Damit wird die erste Forschungsfrage dieser Arbeit beantwortet und durch die Ableitung der Maßnahmen eine Übersicht für die Konzeption von Unterstützungsmaßnahmen geschaffen:

- FF1: Welche Probleme existieren für Studierende in der Modellierung mit der UML?

5.1 ÜBERSICHT

Der folgende Abschnitt führt das Studiendesign zum empirischen Nachweis der existierenden Probleme Studierender **während** des Modellierungsprozesses ein und gibt einen kurzen Überblick zur wissenschaftstheoretischen Einordnung. Für das Design der Studie, wurde zunächst der wissenschaftstheoretische Ansatz festgelegt. Es existieren drei Formen von Erhebungen: Qualitative, quantitative Studien und Mixed–Methods Ansätze. Alle drei Untersuchungsansätze haben ihre Vor– und Nachteile und sind je nach Forschungsschwerpunkt mehr oder weniger für die weiteren Untersuchungen als Erhebungsmethode geeignet:

- Quantitative Studien werden vorwiegend im naturwissenschaftlichen Bereich angewendet. Das Ziel einer quantitativen Studie ist die Validierung von theoretischen Annahmen. Die Datenerhebung geschieht strukturiert in einer Laborumgebung und die Ergebnisse werden statistisch ausgewertet. Ein wichtiges Kriterium für eine repräsentative, quantitative Studie ist eine umfangreiche Studiengruppe (Döring & Bortz, 2016). Diese Art der Studie erscheint für das Forschungsvorhaben nicht geeignet, da der Fokus der Studie darin besteht, möglichst viele Schwierigkeiten zu identifizieren und zu extrahieren.
- Qualitative Studien legen den Fokus auf die Beantwortung von offenen Forschungsfragen. Die Ergebnisse einer qualitativen Studie können unerwartet sein und oft werden, erst auf diesen Resultaten aufbauend, theoretische Hypothesen erstellt. Bei einer qualitativen Untersuchung wird weniger Wert auf einen strukturierten Aufbau der Studie gelegt, als auf eine „sehr detaillierte und umfassende

Analyse“ (Döring & Bortz, 2016, S.184). Diese Form der Untersuchung eignet sich für das Extrahieren von Problemen, wie es Ziel dieser Forschungsarbeit ist. Dabei können die auftretenden Schwierigkeiten extrahiert werden, ohne dass die Ergebnisse durch vorformulierte Hypothesen beeinflusst, bzw. eingeschränkt werden.

- Mixed–Methods ist eine Form der Untersuchung, die in unterschiedlichem Grade die Ansätze aus qualitativen und quantitativen Paradigmen vereint. Das Konzept beinhaltet die Durchführung mehrerer Teilstudien, die sich in ihren Ergebnissen aufeinander beziehen. Bei der Verwendung eines Vorstudienmodells dient eine einfache qualitative Vorstudie als Grundlage für die Entwicklung einer quantitativen Studie in einem größeren Rahmen. Im Gegensatz dazu wird bei dem Vertiefungsmodell eine quantitative Studie durchgeführt und im Nachhinein ausgewählte Probanden, beispielsweise im Rahmen eines Leitfadeninterviews (siehe Mayring, 2016), noch einmal zu konkreten Aspekten befragt (Döring & Bortz, 2016). Eine Mixed–Methods Studie eignet sich für das Forschungsziel, um aus der Literatur bekannte Schwierigkeiten von Studierenden bei der Verwendung von UML festzustellen und diese mit Hilfe einer Beobachtungsstudie und begleitenden retrospektiven Interviews zu vervollständigen.

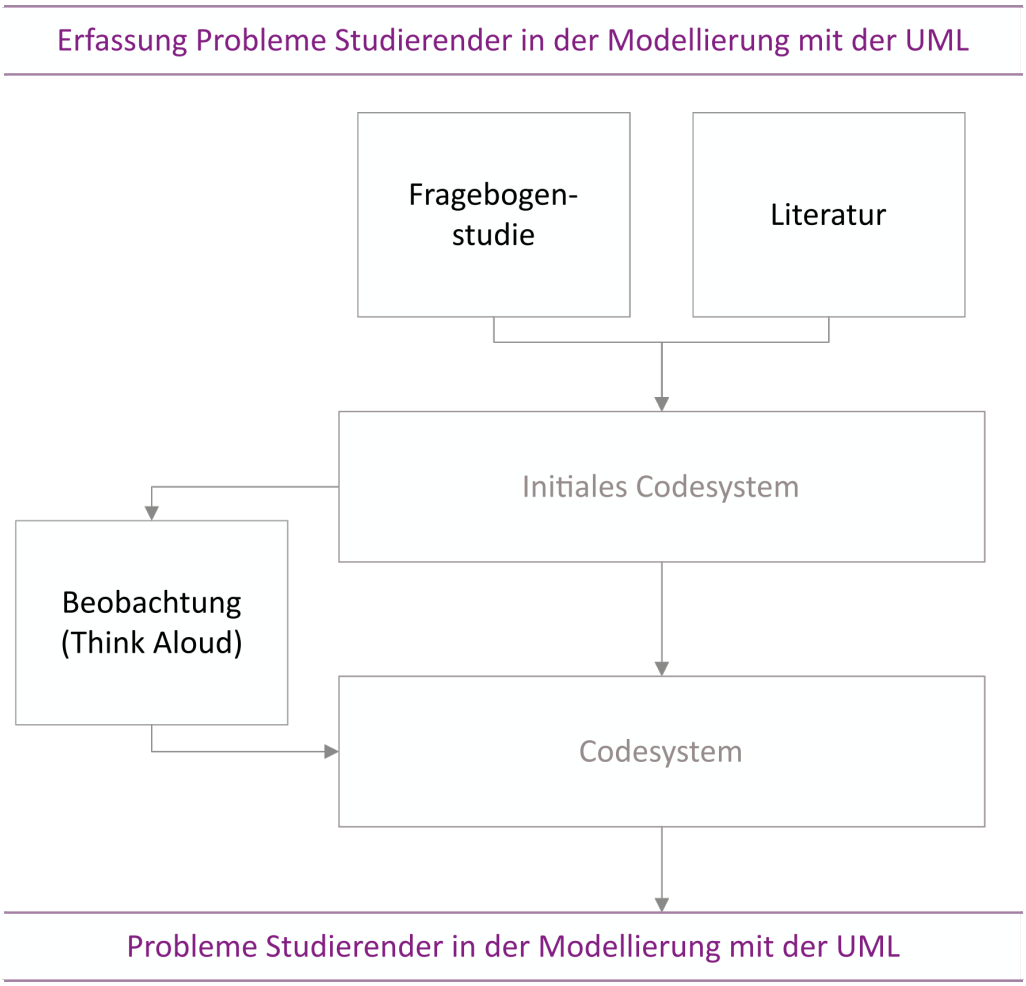


Abbildung 5.1: Erhebung der Probleme Studierender im Modellierungsprozess mit Hilfe der UML

Insgesamt wurde für die Erfassung der Probleme das Mixed-Methods Studiendesign vorgesehen, das in Abb. 5.1 visualisiert ist:

Zunächst wurde eine systematische Literaturrecherche angefertigt, die bereits identifizierte Probleme Studierender erfasst. Zusätzlich zur Literaturrecherche wurden Studierende im Rahmen einer Fragebogenstudie nach aktuellen Schwierigkeiten befragt. Auf Basis dieser Daten wurde ein initiales Codesystem erstellt, das die erfassten Probleme enthielt. Als initiales Codesystem wird hier das Kategoriensystem verstanden, das für die spätere Auswertung verwendet wurde. Dabei wurde festgestellt, dass bisherige Arbeiten nicht den Modellierungsprozess von Studierenden betrachten, diese Arbeit aber nach Unterstützungsmöglichkeiten für ebendiesen sucht. Deshalb wurde eine Studie durchgeführt, die Studierende während der Modellierung eines Softwaresystems beobachtet und eventuelle Schwierigkeiten identifiziert, die sich entweder mit den bereits existierenden Schwierigkeiten aus Online-Fragebogenstudie und

Literatur decken oder dem entwickelten Codesystem Neue hinzufügen. Das letztendlich entstandene Codesystem entspricht einem Katalog über die existierenden Probleme und damit dem ersten Inkrement laut DSR, das als Basis für die Implementierung der Software dient.

Die unten folgenden Ausführungen finden sich ebenfalls in verkürzter Darstellung in Reuter et al. (2020)[16].

5.2 Wissenschaftliche Ausgangslage: Literaturrecherche

Für die Konsultation der Wissensbasis wurde ähnlich zu dem von Kitchenham und Charters (2007) vorgeschlagenen Vorgehen zur Erstellung von systematischen Literaturreviews (SLR) im Software Engineering gehandelt, da diese Arbeit auch in dieser Disziplin anzusiedeln ist.

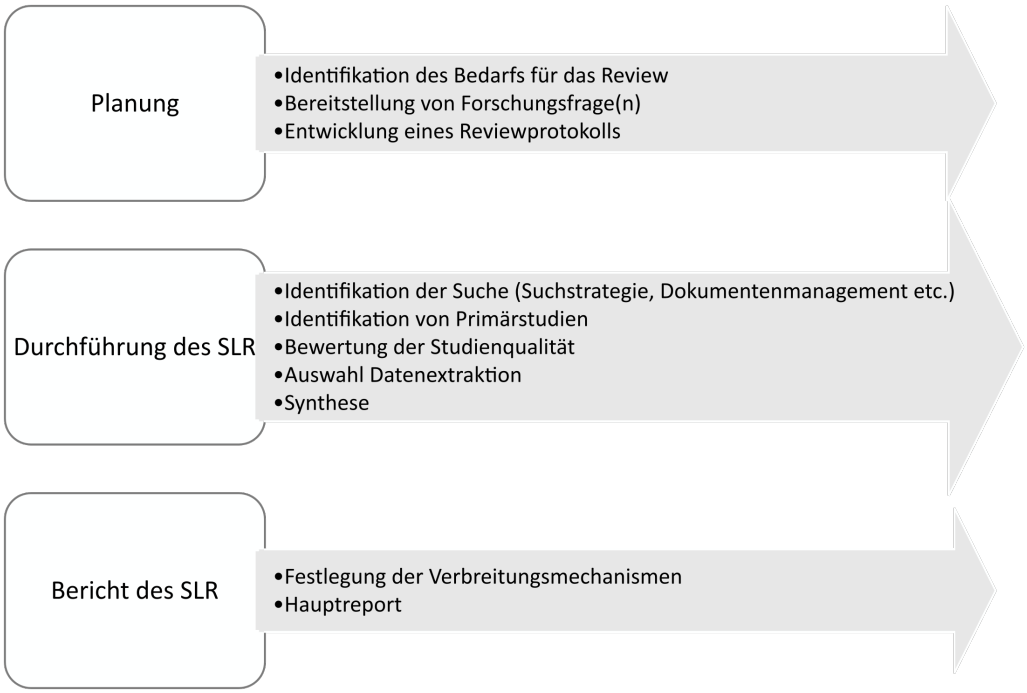


Abbildung 5.2: Review Prozess (adaptiert von Kitchenham und Charters (2007))

Kitchenham und Charters (2007) schlagen einen drei-stufigen Prozess dazu vor (Abb. 5.2), der so auch verfolgt wurde. Das Ziel und die Forschungsfrage, die diesem Systematisches Literatur Review (SLR) zu Grunde liegt, sind oben bereits benannt worden (FF1):

Welche Probleme existieren für Studierende in der Modellierung?

Ein Protokoll wurde, wie von Kitchenham und Charters (2007) vorgeschlagen, mit folgenden Kriterien angelegt und beschrieben:

16 ©2020 IEEE

- Grund für das Review (s.o.)
- Forschungsfrage (s.o.)
- Suchstrategien
  - Suchterme
  - Quellen
  - Ablagemanagement
- Auswahlstrategien
  - Inklusionskriterien
  - Exklusionskriterien
    - Qualitätskriterien
    - Extraktion relevanter Daten

5.2.1 *Suchstrategien*

Die Lehre zu objektorientierter Analyse und Design von Software ist nicht auf einen Sprachraum begrenzt, weshalb deutsch- und vor allem aber englischsprachige Literatur berücksichtigt wird.

Die Literaturrecherche wurde computerbasiert durchgeführt und fand im letzten Quartal 2018 statt. Berücksichtigt wurden die in den Disziplinen einschlägig bekannten Datenbanken:

- ACM
- IEEE
- Sagejournals
- Scopus
- Wiley
- Sciencedirect
- ERIC
- Springer
- Web of Science
- Google Scholar wurde aufgrund undurchsichtiger Suchalgorithmen nur als Meta-Datenbank (gemeint ist, dass sie wieder auf Datenbanken referenziert) konsultiert, wird aber nicht explizit aufgeführt.

Sofern vorhanden, wurde für die einzelnen Suchanfragen ein erweiterter Suchmodus genutzt, um den Suchterm verwenden zu können. Der Suchterm musste an die Abfragesprache der jeweiligen Datenbank angepasst werden. Dabei wurde lediglich die Syntax und die Semantik des Terms verändert, damit eine gemeinsame Betrachtung in der späteren Analyse ohne Einschränkungen durchgeführt werden kann. Mit folgendem Suchterm, abgeleitet aus der Forschungsfrage (FF1) (Brereton,

Kitchenham, Budgen, Turner & Khalil, 2007, 4) und unter Berücksichtigung von Synonymen, wurde gesucht:

(difficulties OR problems) AND (teaching OR learning) AND („modeling software“ OR „modeling software“ OR „software design“ OR „software modeling“ OR „software architecture“ OR „software modeling“) AND UML

Für das Ablagemanagement wurde Mendeley ausgewählt. Das Material wurde exportiert und nach Datenbanken sortiert abgelegt. Dort wurde ebenfalls die Anwendung der Inklusionskriterien/- und Exklusionskriterien mittels Tags vorgenommen.

5.2.2 *Auswahlstrategien*

Um für die Forschungsfrage relevante Informationen (konkrete Probleme Studierender bei der Modellierung mit der UML) zu erhalten, wurde das Material anhand von **Inklusionskriterien** gefiltert. Folgende formale Inklusionskriterien führten dazu, dass das Material als relevant betrachtet wurde:

- Das Literaturformat entspricht einem der folgenden Formate:
  - ein Buch (Monographie)
  - ein Journal- oder Magazinbeitrag
  - ein Beitrag in einem Tagungsband
  - Konferenzband oder Sammelwerk
  - eine Dissertation
  - ein Report
- Das Material ist in Englisch oder Deutsch verfasst.
- Das Material enthält (in Titel und/oder Abstract) Textsegmente, die darauf hindeuten, dass in dem Material, **Probleme** oder **Fehler** erwähnt werden, die bei der **Modellierung** mit der **UML** oder in UML Artefakten entdeckt wurden.
- Das Material identifiziert Probleme bei der Modellierung mit der UML.
- Das Material setzt Studierende in den Fokus.

Aufgrund der folgenden inhaltlichen **Exklusionskriterien** wird Material von der Ergebnismenge ausgeschlossen:

- Das Material beschreibt keine Probleme, die bei der Modellierung mit der UML bei Studierenden auftreten.
- Das Material enthält lediglich eine Beschreibung, die Schwierigkeiten mit der Modellierung als gegeben annimmt.

- Das Material enthält keine Beschreibung zur Methodik wie Probleme identifiziert werden.
- Das Material beschreibt Wege zur Verifikation der Korrektheit von UML-Modellen.
- Das Material beschreibt Lehrkonzepte für die Modellierung mit der UML ohne konkret Probleme Studierender zu benennen.

Für das initiale Codesystem, das ein Ziel des Reviews war, wurden die im Material konkret benannten Probleme in tabellarischer Form (betroffenes Diagramm, Problem, Quelle) erfasst. Material, das weiterführende Probleme identifiziert, findet sich nicht in der tabellarischen Aufstellung, da zunächst diagrammspezifische Probleme im Vordergrund stehen. Dieses wird gesondert gesammelt und in der Auswertung textuell beschrieben. Es gibt also Material, das als „related“ identifiziert wurde, und/oder Material, das direkten Einfluss auf das initiale Codesystem hat. Beide Arten sind im Bericht berücksichtigt und es wurde vermerkt, wenn ein Einfluss auf das Codesystem bestand.

5.2.3 *Durchführung des Reviews*

Für die Durchführung des Reviews wurde nach dem Ablaufmodell in Abb. 5.3 vorgegangen. Das Review wurde zeitgleich auch im Rahmen einer Masterarbeit (Müller, 2019) durchgeführt, sodass die Auswahl des Materials in Diskussion getroffen wurde.

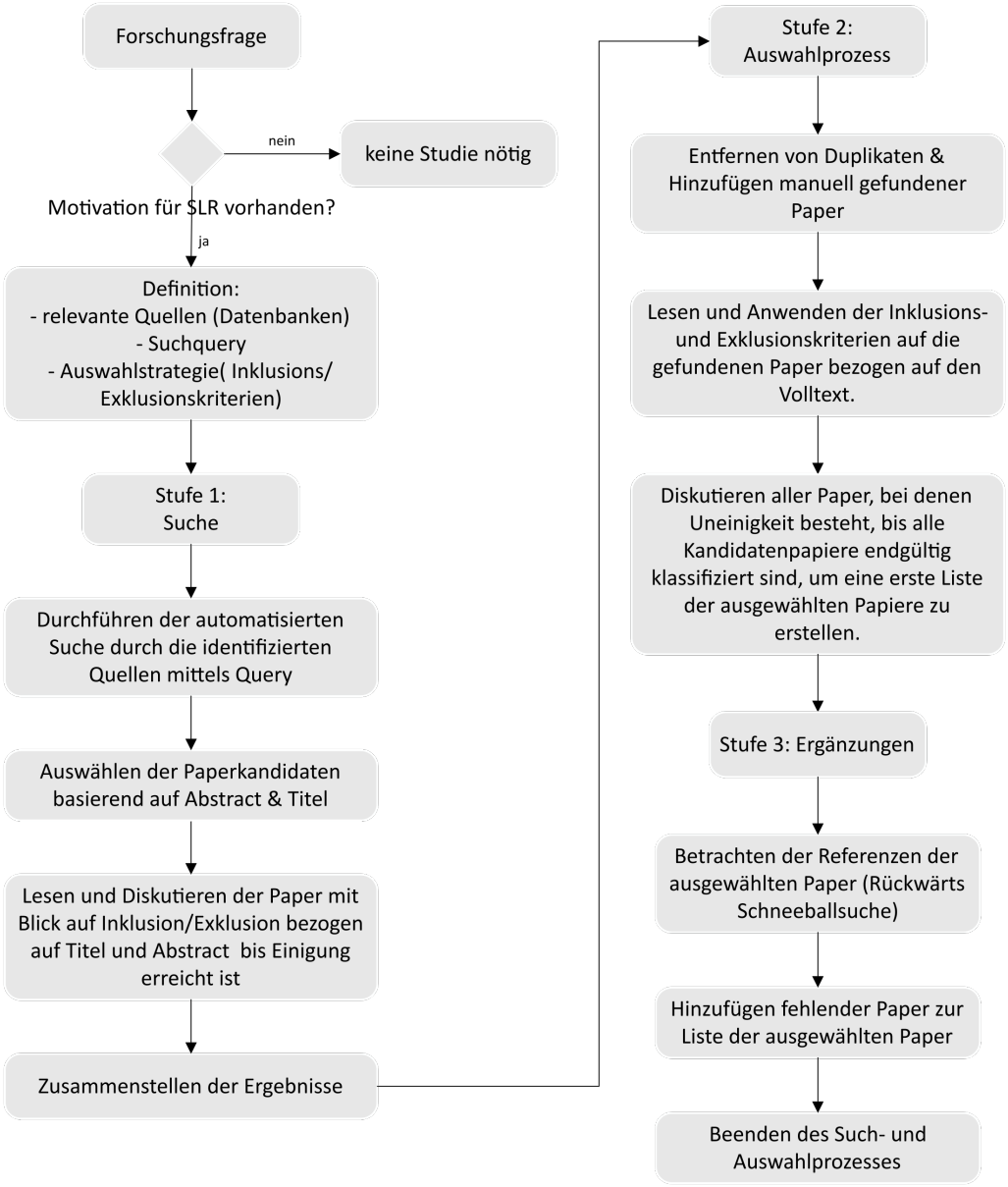


Abbildung 5.3: Ablaufschema zum Vorgehen bei der Literaturrecherche (angelehnt an Kitchenham und Brereton (2013))

Nach der informellen Suche wurden die oben genannten Datenbanken als relevante Quellen festgelegt. Im Anschluss fand dann die automatisierte, computergestützte Suche mit dem Suchterm in den Datenbanken statt. Insgesamt wurde das in der folgenden Tabelle dargestellte Ergebnis (Spalte 1) über die verschiedenen Datenbanken erzielt:

Tabelle 5.1: Durchsuchte Datenbanken und die Summe der identifizierten Publikationen in Spalte 1 sowie die Anzahl der identifizierten Veröffentlichungen nach Filterung nach Titel sowie Abstract in Spalte 2

Datenbanken	# Gefundene Paper	# Nach Titel & Abstract
ACM	20	9
ERIC	1	1
IEEE	15	6
Sagejournals	154	45
Scopus	42	19
Wiley	1	0
Sciencedirect	52	3
Springer	0	0
Web of Knowledge	3	1

Die Bewertung des Materials hinsichtlich Titel und Abstract wurde gemeinsam mit einer Masterandin (Müller, 2019) vorgenommen, die diese Literaturrecherche, parallel, als Teil der Masterarbeit vorgenommen hat. Spalte 2 zeigt die Ergebnisse der Iteration nach Anwendung der Inklusionskriterien und Exklusionskriterien auf Titel und Abstract des Materials. Mit diesen Ergebnissen wurde Stufe 2 gestartet. Duplikate wurden entfernt und Experten wurden nach weiterer Literatur befragt. Eine weitere tabellarische Darstellung erfolgt nicht. Da identische Publikationen in mehreren Datenbanken vorkamen, verschwimmt die Zuordnung der Publikation zur Datenbank. Es ergaben sich $N = 50$ Publikationen zur weiteren Analyse. Diese wurden dann codiert und unter Anwendung der Inklusionskriterien und Exklusionskriterien $N = 12$ Publikationen als relevant herausgearbeitet. Abschließend wurde eine rückwärtsgerichtete Schneeballsuche durchgeführt, jedoch kein weiteres Material mehr identifiziert.

5.2.4 *Bericht des Reviews*

Die folgenden Publikationen beschreiben das Ergebnis der Literaturrecherche sowie Einflüsse und Zusammenhänge mit der Erhebung, die im Nachgang beschrieben werden.

Duschl, Obermeier und Vogel-Heuser (2014) untersuchen die Fehler von Studierenden und deren Ursachen in Klassen- und Zustandsdiagrammen im Bereich der Anlagenautomatisierung. Sie evaluieren ebenfalls die

Artefakte der Studierenden und befragen Studierende zudem nach möglichen Ursachen auf der Basis des generischen Fehlermodellierungssystems (GEMS) von Reason (1991). Leider werden die Bewertungskriterien für die Artefakte nicht vorgestellt, so dass keine Rückschlüsse auf die spezifischen Fehler gezogen werden können.

Interessante Erkenntnisse sind jedoch die Ursachen für die Probleme der Studierenden:

- Zeitmangel
- Übersetzungsproblem (Translation Problem, Studierende können das, was sie modellieren wollen nicht in das Modell übertragen)
- Ablenkung
- Missverständnisse
- Übersehen
- fehlerhafte Aufgabenbeschreibungen
- vorzeitiger Abbruch

Aus dieser Arbeit können keine fachlichen Probleme Studierender mit der UML abgeleitet werden, allerdings können Ähnlichkeiten festgestellt werden zum sogenannten Translation Problem, d. h. der Übertragung des mentalen Modells auf das UML-Modell. Duschl et al. (2014) stellen dieses Problem als die zweithäufigste Ursache dar. Dieses Problem geht ebenfalls in das initiale Codesystem mit ein.

Akayama et al. (2013) stellen eine Lernumgebung für modellgetriebene Entwicklung vor. Sie präsentieren Lernziele in Bezug auf Klassen- und Zustandsdiagramme, ein grundlegendes Verständnis des Modellierungsprozesses und die Notation für statische und dynamische Modellierung. Im Hinblick auf das Verständnis der Lernmodellierung nennen die Autoren drei Aspekte:

- Es fehlt das grundlegende Bewusstsein für die Notwendigkeit der Modellierung.
- Studierende finden nicht die richtige Abstraktionsebene für ein Modell.
- Die Validierung von Modellen erfordert deren Implementierung oder eine Überprüfung durch einen Dozierenden, weshalb die Studierenden Probleme bei der Verifizierung ihrer Modelle haben.

Akayama et al. (2013) nennen keine speziellen auf einzelne Diagramme bezogene Probleme, jedoch sind für die Arbeit gerade die beiden letztgenannten Probleme von Interesse. sie werden in das initiale Codesystem aufgenommen.

Stikkolorum, Gomes De Oliveira Neto und Chaudron (2018) evaluieren didaktische Ansätze, genauer gesagt Interventionen durch Dozierende,

bezeichnet als Teaching Assistants (TA), bei Analyse und Design mit der UML. Untersucht werden auch typische Herausforderungen, die im Umgang mit diesem Thema eine Rolle spielen und für die die Studierenden Hilfe bei den TAs suchen. Als Erhebungsinstrumente wurden sowohl Interviews mit den TAs als auch Fragebögen und studentische Artefakte verwendet. Es wurden Use-Case-Diagramme, Klassendiagramme, Sequenzdiagramme und Zustandsdiagramme behandelt. Von besonderem Interesse für die Forschung ist der Teil der Arbeit, der nach den Schwierigkeiten der Studierenden fragt, die mit den Assistenten besprochen wurden. Diese Erkenntnisse werden ebenfalls als Grundlage für das Codesystem verwendet.

Ausgangspunkt für Bolloju und Leung (2006) ist die Behauptung von Schenk, Vitalari und Davis (1998), dass Anfänger, die konzeptuelle Modelle entwickeln, größere Schwierigkeiten haben als erfahrene Analytiker, was das domänenspezifische Wissen, die Problemstrukturierung und die kognitiven Prozesse betrifft. Obwohl es bereits Empfehlungen und Richtlinien gibt, die die Verwendung von Patterns vorschlagen, stellen sie fest, dass Anfänger nicht den maximalen Nutzen aus dieser Unterstützung ziehen können. Der Grund dafür sei die kognitive Überlastung, die mit den Empfehlungen und Richtlinien einhergehe (Bolloju & Leung, 2006). Diese Grundlage entspricht weitgehend den Ansichten, die in dieser Arbeit unterstützt werden (siehe Abschnitt 3.2.1. Kognitive Überlastung durch vorhandenes Arbeitsmaterial könnte eine Ursache für Probleme Studierender sein. In ihrer Arbeit analysieren sie typische Fehler in Artefakten von Novizen: Use-Case-Diagramm, Anwendungsfallbeschreibung, Klassendiagramm und Sequenzdiagramm. Die Fehler wurden auf der Grundlage eines bestehenden semiotischen Rahmens (Lindland, Guttorm & Solvberg, 1994) in drei Kategorien eingeteilt: syntaktisch, semantisch und pragmatisch. Für die zugrundeliegende Forschung sind sowohl die Klassifikation als auch die einzelnen gefundenen Fehler interessant. Die einzelnen Probleme werden in das initiale Codesystem aufgenommen.

Siau und Loo (2006) verwenden die Concept-Mapping-Technique (Trochim (1989)) in einem Kurs, der sich auf die Einführung in die Konzepte, Syntax, Semantik und Diagrammtechnik der UML konzentriert, um die Schwierigkeiten der Studierenden mit diesen Themen herauszufinden. Mit Hilfe dieser Methode leiten sie 15 Cluster ab und kategorisieren diese wiederum in fünf Meta-Regionen, die die von den Studierenden als besonders schwierig empfundenen Probleme darstellen:

- Schulungsmaterial
- Vorkenntnisse
- UML-Diagramme
- UML-Semantik

- UML-Konstrukte

In einem weiteren Schritt konsolidieren sie die Meta-Regionen in inhärente und periphere Kategorien. Die erste Kategorie umfasst Probleme im Zusammenhang mit UML-Diagrammen, UML-Semantik und UML-Konstrukten. Kategorie zwei konsolidiert periphere Probleme beim Lernen von UML im Zusammenhang mit Schulungsmaterial/Software und dem Vorhandensein oder Fehlen von Vorkenntnissen. Die konkreter bezeichneten Probleme wurden in das Codesystem aufgenommen.

Sien et al. (2010) untersucht die Schwierigkeiten und Missverständnisse Studierender mit Klassen- und Sequenzdiagrammen. Sien stellt fest, dass insbesondere Klassendiagramme Probleme verursachen, weil Studierende nicht die richtige Abstraktionsebene finden. Sie wüssten nicht, „was“ sie modellieren sollen. Insbesondere das Klassendiagramm sei jedoch wichtig, da es grundlegend zum Objektmodellierungsprozess beitrage. Die Artefakte Studierender wurden auf Vollständigkeit, Komplexität und Genauigkeit bewertet.

Kruus, Robal und Jervan (2014) beschreiben Probleme von Studierenden bei der Modellierung von Use-Case-Diagrammen, Aktivitätsdiagrammen, Zustandsdiagrammen und Klassendiagrammen. Sie extrahieren die Probleme ihrer Studierenden auf der Grundlage von Feedback[17] und Berichten, die sie pro Abgabe vorlegen mussten. Die Erkenntnisse fließen in das Codesystem ein.

VanderMeer und Dutta (2009) beschäftigten sich mit Problemen der kognitiven Komplexität, die bei der Modellierung von Sequenzdiagrammen auftreten, und vor allem mit möglichen Ursachen für diese. Darüber hinaus geben sie, basierend auf dem LCD (Learning Centered Design)-Rahmenwerk von Reeves (1999), Empfehlungen für die Präsentation von Aufgaben, um die Lernkomplexität für die Studierenden zu reduzieren. Hinsichtlich bereits bekannter Arbeiten zu Problemen von Studierenden beziehen sie sich auf Bolloju und Leung (2006), und Siau und Loo (2006). Diese Arbeit ist wegen der Hinweise („Engage the Learner Actively“, „Use Examples and Exercises Extensively“) auf mögliche Scaffolds[18] für Sequenzdiagramme interessant.

Thomasson, Ratcliffe und Thomas (2006b) untersuchen die Schwierigkeiten von Novizen beim Entwerfen von Klassendiagrammen. Sie analysieren deren Entwürfe auf Fehler und fanden heraus, dass nicht referenzierte Klassen und Probleme mit Attributen, und Klassenkohäsion die häufigsten Probleme sind. Die Probleme werden extrahiert, indem die Ergebnisse der Studierenden auf Vollständigkeit und Eignung hin

17 Neben dem Artefakt sollten die Studierenden dort auch ihre Probleme in einem Präsentationsbericht beschreiben.

18 Genauere Informationen dazu finden sich in späteren Kapiteln der Arbeit.

bewertet werden. Die Ergebnisse der Studie werden in das Codesystem aufgenommen.

Or-Bach und Lavy (2004) analysierten Klassendiagramme von Studierenden im Hinblick auf mögliche Abstraktionsprobleme. Als grundlegende Voraussetzung für die Abstraktion legen sie die Definition einer Klasse mit ihren relevanten Attributen und Methoden fest. Ihren Erkenntnissen zufolge, haben die meisten Studenten diese Stufen noch nicht verstanden. Sie definieren die Verwendung einer Klassenhierarchie mit einer abstrakten Klasse als eine komplexere Abstraktionsaufgabe und erstellen darauf aufbauend eine 3-stufige Taxonomie: Ebene 1 wird als abstrakte Klasse beschrieben, die Attribute enthält, Ebene 2 enthält die abstrakte Klasse und Attribute sowie implementierte Methoden, auf der höchsten Ebene (3) sind zusätzlich abstrakte Methoden enthalten. Nur wenige Studierende erreichen Or-Bach und Lavy (2004) zufolge dieses Niveau.

Holland, Griffiths und Woodman (1997) stellt in seinem Werk falsche Vorstellungen über objektorientierte Konzepte bei Novizen vor. In dieser Arbeit fehlen empirische Ansätze und eine Methodik zur Extraktion und Analyse von Missverständnissen, weshalb sie zwar als verwandt betrachtet werden kann, aber nicht in das Codesystem aufgenommen wird.

Chren, Buhnova, Macak, Daubner und Rossi (2019) stellen die für diesen Teil der Arbeit relevanteste Publikation vor. Sie analysierten Einreichungen von Studierenden zur Verwendung von Use-Case-Diagrammen, Aktivitätsdiagrammen, Klassendiagrammen in zwei Varianten (analytisches Klassendiagramm und Design-Klassendiagramm), Zustandsautomaten, Sequenzdiagrammen, Kommunikationsdiagrammen und Entity-Relationship-Diagrammen (nicht UML). Sie leiteten ähnlich zu dem geplanten Vorhaben auch einen Katalog von Problemen ab und identifizierten 146 Fehler[19] in Diagrammen von Studierenden. Die Analyse der Diagramme fand ähnlich wie hier, auf Basis von bereits existierenden Problemen aus der Literatur und praktischem Wissen von Experten statt. Der gesamte Katalog liegt zum Zeitpunkt der Erstellung der Arbeit allerdings in tschechischer Sprache vor und ist nur unvollständig in englischer Sprache vorhanden. Zudem wurde diese Arbeit erst nach Durchführung und Auswertung der Beobachtungsstudie veröffentlicht, sodass sie nur zur Vervollständigung herangezogen werden kann, da das initiale Codesystem zum Zeitpunkt der Veröffentlichung der Arbeit schon bestand.

19 Es bleibt unklar, ob hier Fehler im Sinne des UML- Metamodells und/oder Fehler im Sinne eines Verstoßes gegen Modellierungsrichtlinien betrachtet wurden.

Die Tabellen in Abschnitt 5.3.3.2 fassen die aus der Literatur exzerpierten, diagrammspezifisch identifizierten Probleme zusammen und stellen gleichzeitig einen Teil des initialen Codesystems für die Erhebung dar. Die folgenden Tabellen zeigen, welche Publikation, welchen Diagrammtyp betrachtet hat (Tabelle 5.2) und zeigt zudem welche Methodik zur Erhebung verwendet wurde (Tabelle 5.3). Auffällig dabei ist, dass keine der relevanten Publikationen den Modellierungsprozess untersucht hat.

Tabelle 5.2: Zuordnung der identifizierten Publikationen zu untersuchter/untersuchten Diagrammart(en), mit „x“ markierte Zellen kennzeichnen die untersuchte(n) Diagrammart(en)

Publikation	**Diagrammart**				
	Use-Casediagramm	Klassendiagramm	Aktivitätsdiagramm	Zustandsdiagramm	Sequenzdiagramm
Duschl, Obermeier und Vogel-Heuser (2014)		x		x	
Akayama et al. (2013)		x		x	
Stikkolorum, Gomes De Oliveira Neto und Chaudron (2018)	x	x	x	x	x
Bolloju und Leung (2006)	x	x			x
Sien et al. (2010)		x			x
Kruus, Robal und Jervan (2014)	x	x	x	x	
VanderMeer und Dutta (2009)					x
Thomasson, Ratcliffe und Thomas (2006b)		x			
Or-Bach und Lavy (2004)		x			
Holland, Griffiths und Woodman (1997)		x			

Tabelle 5.3: Zuordnung der identifizierten Publikationen zu verwendeten Erhebungsmethoden, mit „x“ markierte Zellen kennzeichnen die verwendete(n) Erhebungsmethode(n)

Publikation	**Erhebungsmethode**			
	Artefaktanalyse	Interviews	Fragebogen	Modellierungsprozess
Duschl, Obermeier und Vogel-Heuser (2014)	x	x	x	
Akayama et al. (2013)				
Stikkolorum, Gomes De Oliveira Neto und Chaudron (2018)	x	x	x	
Bolloju und Leung (2006)	x			
Sien et al. (2010)	x			
Kruus, Robal und Jervan (2014)	x			
VanderMeer und Dutta (2009)	x			
Thomasson, Ratcliffe und Thomas (2006b)	x			
Or-Bach und Lavy (2004)	x			
Holland, Griffiths und Woodman (1997)	x			

5.3 EMPIRISCHE ERHEBUNGEN ZU PROBLEMEN STUDIERENDER MIT DER UML

Wie bereits einleitend angeführt, wurde eine eigene Studie zusätzlich zur systematischen Literatursuche durchgeführt, da keine der relevanten Publikationen direkt den Modellierungsprozess der Studierenden betrachtet hat, sondern ausschließlich Artefakte Studierender analysiert wurden.

Um mögliche während des Modellierungsprozesses auftretende Probleme zu erfassen, wurde eine zweigeteilte Erhebung durchgeführt. Die Studierenden füllten pro Diagrammtyp einerseits eine Online-Umfrage bestehend aus vier Fragen aus[20] und andererseits wurde eine Think-Aloud- Beobachtungsstudie durchgeführt. Beide Formate werden in den folgenden Abschnitten näher beschrieben.

Die Erhebung fand an zwei bayerischen Hochschulen innerhalb des Projekts EVELIN statt. Beide Kurse, in welchen die Daten erhoben wurden, waren entsprechend der Projektausrichtung Software Engineering Kurse und für Novizen konzipiert. Das bedeutet, dass die Studierenden zum ersten Mal mit der Modellierung eines Softwaresystems konfrontiert waren. Um eine möglichst hohe Kontrastierung zu erreichen, wurde ein Software Engineering Kurs für Informatiker und ein Kurs für Nicht-Informatiker ausgewählt (gemeint sind Studierenden aus Ingenieurstudiengängen, die nur einen geringen Anteil an Informatik-Modulen in ihrem Studiengang studieren). In beiden Kursen werden die folgenden Diagrammarten gelehrt:

- Use-Case-Diagramm
- Klassendiagramm
- Aktivitätsdiagramm
- Zustandsdiagramm
- Sequenzdiagramm

Nach Frezza und Andersen (2006), Granda, Condori-Fernandez, Vos und Pastor (2015), Anke und Bente (2019), gehören diese Diagrammtypen zu den am häufigsten eingesetzten Diagrammtypen in der industriellen Praxis.

Das Kursmodul besteht - in beiden Fällen - aus einer Vorlesung und einer vorlesungsbegleitenden Übung. Das Experiment wurde vollständig in die Lehreinheiten integriert, d. h. die Studierenden hörten sich die Vorlesung(en) über UML-Modellierung an und besuchten die zugehörige Übung, in der das Experiment stattfand.

20 Zwei der vier Items betreffen dabei gewünschte Hilfsmittel bzw. offene Fragen der Studierenden, zwei Items betreffen existierende Probleme. Deshalb werden zunächst nur die entsprechenden zwei Items diskutiert.

Der Kurs für die Nicht-Informatik-Studiengänge hatte zwei Vorlesungen, denen jeweils eine Übung folgte. In der ersten Vorlesung wurden Use-Case-Diagramme und Klassendiagramme vorgestellt, in der zweiten Vorlesung Zustandsdiagramme und Sequenzdiagramme. Die Übung für UML-Inhalte knüpft thematisch an die vorhergehende Phase und an die Übung zum Requirements Engineering an: Es ging um einen Kaffeevollautomaten, für den eine Software entwickelt werden soll. Verschiedene Kaffeemaschinenhersteller und andere Beteiligte haben unterschiedliche Anforderungen an dieses Produkt und die zugehörige Software (Artefakt der vorherigen Übung), die zunächst in Anwendungsfällen modelliert werden sollten. Anschließend sollte ein Grobentwurf auf der Grundlage des Anforderungsdokuments mit Hilfe eines Klassendiagramms entworfen werden. Der Zustandsautomat und das Sequenzdiagramm wurden mit Hilfe von detaillierten Aufgabentexten und Vorgaben zu den Diagrammen eingeführt und mussten bis vervollständigt werden. Die Studierenden konnten zwischen Astah und Enterprise Architect als Modellierungsumgebung wählen. Die meisten Studenten wählten Enterprise Architect, da Astah auf den Computern der Hochschulen nicht vorinstalliert war.

Die Übung für die Informatikstudierenden hatte eine etwas andere Kursstruktur. Es handelte sich um eine vierwöchige vorlesungsbegleitende Veranstaltung, die sich in dieser Zeit ausschließlich mit der Modellierung mit UML beschäftigte. Zu Beginn des Praktikums erhielten die Studierenden ein Szenario und die Aufgaben für das gesamte Praktikum. Ziel war es, Software für eine Carsharing-Agentur zu entwickeln. Zuerst sollte ein Use-Case-Diagramm entworfen werden, gefolgt von einer textuellen Beschreibung eines der Anwendungsfälle. Für diesen Anwendungsfall sollte ein Aktivitätsdiagramm modelliert werden. Teil vier erforderte die Erstellung eines Klassendiagramms, in dem Attribute, aber keine Methoden modelliert werden sollten. Die nächste Teilaufgabe befasste sich mit Zustandsdiagrammen und der Modellierung des Lebenszyklus einer Fahrzeugreservierung. Schließlich wurde noch einmal auf den textuell spezifizierten Anwendungsfall Bezug genommen und ein Sequenzdiagramm sollte erstellt werden. Als Modellierungsumgebung konnten die Studierenden zwischen Modelio, IBM Rational Software Architect und Enterprise Architect wählen. Die meisten Studierende hatten sich aufgrund der kostenfreien Nutzung für Modelio entschieden.

5.3.1 *Wahl der Erhebungsmethoden*

Im Folgenden wird nun der Entscheidungsprozess für die Erhebungsmethoden dargelegt. Dabei wird Döring und Bortz (2016) als zentrale

Quelle herangezogen, da diese als wissenschaftlich anerkannt gilt und eine wohl definierte Auswahl an Methoden bietet. Döring und Bortz (2016) unterscheiden sechs Datenerhebungstechniken: die Beobachtung, das Interview, die schriftliche Befragung bzw. die selbstadministrierte Fragebogenmethode, den psychologischen Test, die physiologische Messung und schließlich die Dokumentenanalyse (Döring & Bortz, 2016, S.322). Mit Ausnahme der physiologischen Messung, gibt es alle Erhebungsmethoden in qualitativen und quantitativen Varianten. Qualitative Datenerhebungstechniken (im Gegensatz zu quantitativen hochstandardisierten Erhebungsmethoden) lassen dem Forscher Raum, situationsspezifisch zu reagieren und beispielsweise mehr oder weniger Details mit in die Datenerhebung mit einzubeziehen und/oder zu entscheiden, welche Aspekte, welchen Stellenwert für das zu untersuchenden Phänomen haben. Durch die Offenheit des Vorgehens „soll eine optimale Annäherung an den Untersuchungsgegenstand gewährleiste[t] [werden] und die Chance erhöh[t] [werden], dass sich auch neue und unerwartete Inhalte in den erhobenen Daten zeigen“ (Döring & Bortz, 2016, S.322).

Die beschriebenen Eigenschaften der qualitativen Erhebungsformate sind notwendig, um das Ziel der Erhebung zu erreichen, nämlich die im Feld auftretenden Probleme, aus Fremdperspektive (also nicht aus Studierendensicht), zu manifestieren und gegebenenfalls um noch nicht identifizierte Probleme zu erweitern.

In einem nächsten Schritt werden die von Döring und Bortz (2016) beschriebenen qualitativen Datenerhebungstechniken und deren Indikatoren[21] kurz vorgestellt und hinsichtlich des Studienziels bewertet, um anschließend die gewählten Erhebungsmethoden in ihrer Anwendung zu beschreiben und abzugrenzen.

5.3.1.1 *Beobachtung*

Laut Döring und Bortz (2016) ist eine wissenschaftlichen Beobachtung („scientific observation“) „die zielgerichtete, systematische und regelgeleitete Erfassung, Dokumentation und Interpretation von Merkmalen, Ereignissen oder Verhaltensweisen mithilfe menschlicher Sinnesorgane und/oder technischer Sensoren zum Zeitpunkt ihres Auftretens“ (Döring & Bortz, 2016, S.324).

Indikatoren:

- Verbal weniger auskunftsfähige Untersuchungsobjekte können gut beobachtet werden (Döring & Bortz, 2016, S.324). Dieser Indikator

21 Döring und Bortz (2016) stellen in den definierten Indikatoren auch häufig Vorteile der Erhebungsmethode im Gegensatz zu einer weiteren Erhebungsmethode vor.

trifft nicht explizit zu. Allerdings sollten Studierende möglichst wenig in ihrem „normalen“ Übungssetting beeinträchtigt werden.

- „Über automatisierte und unbewusste Verhaltensweisen [...] lassen sich mittels Beobachtung, nicht jedoch mittels Befragung Erkenntnisse gewinnen“ (Döring & Bortz, 2016, S.325). Da vermutet wurde, dass sich Studierende nicht all ihrer Probleme bewusst sind und sie deshalb nicht äußern oder benennen können, ist dieser Indikator gegeben.
- „Auch Themen oder Zielgruppen, bei denen die Auskunftswilligkeit beschränkt ist, können mittels Beobachtung untersucht werden. Geringe Auskunftswilligkeit besteht nicht nur bei normverletzenden Verhaltensweisen, sondern oft schlicht aus Zeitmangel (z. B. erklären sich viel beschäftigte Fachkräfte manchmal eher bereit, sich bei der Arbeit beobachten zu lassen als an Interviews über ihren Arbeitsalltag teilzunehmen, die einen zeitlichen Zusatzaufwand bedeuten würden)“ (Döring & Bortz, 2016, S.325). Bezogen auf die Studierendengruppen trifft dieser Indikator auch zu, da diese häufig ebenso keine zusätzliche Zeit haben oder aufwenden wollen. Eine Aufnahme während des Zeitraums der Übung verhindert dies.
- „Die Befragung ist immer eine reaktive Methode, die auf die aktive und bewusste Mitwirkung der Befragten angewiesen ist und somit Daten produziert, die mehr oder minder deutliche Verzerrungen (z. B. durch Selbstdarstellungsverhalten) beinhalten. Eine Beobachtung kann dagegen non-reaktiv (d. h. nicht- teilnehmend und verdeckt) stattfinden, so dass nicht in die natürlichen Abläufe eingegriffen wird (z. B. erfragte Selbstauskunft zum umweltbewussten bzw. abfallvermeidenden Verhalten versus beobachtete tatsächliche Menge und Zusammensetzung des Hausmülls; Selbstauskunft des Service-Personals zur Qualität der Kundenbetreuung versus Beobachtung der Dienstleistung)“ (Döring & Bortz, 2016, S.325). Ziel der Studie ist auch die Erfassung der Probleme im natürlichen Umfeld, ohne dass Probleme erinnert werden müssen, wie beispielsweise in einer Befragung retrospektiv oder ein konstruiertes Setting erschaffen wird.
- „Nicht zuletzt kann die Befragung immer nur zusammenfassende und punktuelle Aussagen über Verhaltensweisen erfassen, während die Beobachtung das Verhalten im Zeitverlauf kontinuierlich abbilden kann (z. B. Veränderungen des Schülerverhaltens über den Verlauf einer gesamten Unterrichtsstunde oder Klassenreise)“ (Döring & Bortz, 2016, S.325). Gerade der Modellierungsprozess sollte im Fokus der Erhebung stehen, da hier derzeit noch keine Probleme analysiert wurden. Deshalb wäre eine punktuelle Befragung der

Studierenden nicht zielführend und würde möglicherweise aufgetretene Probleme, die die Studierenden vergessen zu berichten, nicht betrachten.

5.3.1.2 *Interview*

Döring und Bortz (2016) definieren ein Interview synonym zu einer wissenschaftlichen mündlichen Befragung als "[...] zielgerichtete, systematische und regelgeleitete Generierung und Erfassung von verbalen Äußerungen einer Befragungsperson (Einzelbefragung) oder mehrerer Befragungspersonen (Paar-, Gruppenbefragung) zu ausgewählten Aspekten ihres Wissens, Erlebens und Verhaltens in mündlicher Form"(Döring & Bortz, 2016, S.356).

Indikatoren:

- „Die Interviewtechnik als Befragungsform hat gegenüber der Beobachtung den Vorteil, dass zum einen Aspekte des subjektiven Erlebens zugänglich werden, die generell nicht der Beobachtung zugänglich sind (z. B. Gefühle, Meinungen, Überzeugungen, Glaubensinhalte) und dass zum anderen auch nicht direkt beobachtbare Ereignisse und Verhaltensweisen erfasst werden können (z. B. Erlebnisse, die sich auf private oder in der Vergangenheit liegende Situationen beziehen)“ (Döring & Bortz, 2016, S.356). Da nicht die subjektiven Einschätzungen der Studierenden im Vordergrund stehen, sondern auch Probleme, die sie selbst gar nicht als solche wahrnehmen, ist dieser Indikator nur bedingt gegeben.
- „Gegenüber der schriftlichen Befragung mittels selbst ausgefülltem Fragebogen, die von den Befragungspersonen entsprechende Sehfähigkeit, Konzentration sowie Lese- und Schreibkompetenzen erfordert, handelt es sich bei der Interviewtechnik um eine niedrigschwellige alltagsnahe Methode, mit der auch Befragungspersonen angesprochen werden, für die eine schriftliche Befragung ungeeignet wäre (z. B. Menschen mit eingeschränkten Lese- und Schreibfähigkeiten)“ (Döring & Bortz, 2016, S.356f.). Dieser Indikator ist für die Beantwortung der Fragestellung nicht relevant.
- „Die mündliche Befragung findet in einer Live-Situation statt (meist als persönliches Interview, aber auch als Telefon- oder Online-Interview), während Fragebögen oft zeitversetzt und ohne direkten Kontakt zum Forschungsteam ausgefüllt werden. Dadurch erhält man mittels Interviewtechnik Hintergrundinformationen über die Befragungspersonen und die Befragungssituation und kann die Datenqualität besser einschätzen (z. B. wie zügig wird geantwortet, wo gibt es Rückfragen, wie kooperativ wirkt die Befragungsper-

son etc.)“ (Döring & Bortz, 2016, S.357). Dieser Indikator kann für die Beantwortung der Fragestellung hilfreich sein, da die direkt auftretenden Probleme während des Modellierungsprozesses von Interesse sind und identifiziert werden sollen.

- „Die Live-Situation des Interviews sorgt für eine persönlichere Atmosphäre als das Ausfüllen eines Fragebogens. Dies kann wünschenswert sein, wenn man Befragungspersonen im Zuge einer Studie individuell ansprechen und auf ihre Antworten eingehen möchte. Diese Option wird insbesondere bei qualitativen Interviews ausgeschöpft“ (Döring & Bortz, 2016, S.357). Dieser Indikator spielt für die Problemstellung keine Rolle, da Probleme Studierender in einem möglichst natürlichen Umfeld erfasst werden sollten.
- „Wenn Befragungspersonen sich mündlich anstatt schriftlich äußern, können sie zu einer einzelnen Frage in kürzerer Zeit viel mehr Informationen liefern. Insbesondere ausführliche Schilderungen komplexer Zusammenhänge oder zeitlich lang ausgedehnter Prozesse erhält man von den meisten Menschen nicht schriftlich, sondern nur mündlich im Zuge qualitativer Interviews. Will man eine größere Zahl von Fragen stellen und/oder handelt es sich um kompliziertere Fragen, so ist das Interview der Fragebogentechnik vorzuziehen, da umfangreiche und komplizierte schriftliche Fragebögen auf geringe Akzeptanz stoßen“ (Döring & Bortz, 2016, S.357). Dieser Indikator trifft zu. Die Erfassung der Probleme Studierender könnte auch im Rahmen von möglichst offenen Interviews geschehen. Mit der Durchführung von Interviews während des Modellierungsprozesses würde allerdings aktiv in den Modellierungsprozess eingegriffen und Studierende müssten sich ständig neue Konzentration aufbauen. Zudem würde dann nur die Perspektive der Studierenden betrachtet und der Prozess retrospektiv betrachtet. Ein Fragebogen vernachlässigt außerdem Problemdetails und mögliche Ursachen und Bedingungen.

5.3.1.3 *Fragebogen*

Döring und Bortz (2016) beschreiben den Fragebogen als „die zielgerichtete, systematische und regelgeleitete Generierung und Erfassung von verbalen und numerischen Selbstauskünften von Befragungspersonen zu ausgewählten Aspekten ihres Erlebens und Verhaltens in schriftlicher Form“ (Döring & Bortz, 2016, S.398).

Indikatoren:

- Mit Hilfe einer Befragung lassen sich „Aspekte des subjektiven Erlebens sowie des vergangenen oder privaten Verhaltens erfassen, die

nicht direkt beobachtbar und auch nicht in Verhaltensspuren oder Dokumenten manifestiert sind“ (Döring & Bortz, 2016, S.398). Die Intention der Studie ist es Probleme direkt, wenn sie auftreten zu beobachten, nichtsdestotrotz ist natürlich die subjektive Einschätzung der einzelnen Studierenden von Interesse.

- „Im Vergleich zur Interviewtechnik ist die Fragebogenmethode durch Selbstadministration viel effizienter: In kurzer Zeit können Fragebogenantworten von vielen Befragungspersonen zu sehr vielen Merkmalen gesammelt werden. Es müssen keine Interviewerinnen und Interviewer rekrutiert, geschult und ins Feld geschickt werden. Auch sind viele Menschen eher bereit, einen Fragebogen auszufüllen als einen Interviewtermin zu verabreden und einzuhalten“ (Döring & Bortz, 2016, S.398). Eine große Stichprobe ist für jede Studie von Vorteil. Je mehr Probleme Studierender erfasst werden können, desto genauer lassen sich Aussagen über Probleme Studierender ableiten.
- „Das Ausfüllen eines Fragebogens ist aus Sicht der Befragungspersonen diskreter und anonymer als eine Interviewsituation. Dementsprechend können mittels Fragebogen heikle und intime Themen besser erhoben werden als im Interview“ (Döring & Bortz, 2016, S.398). Diskretion und Anonymität ist gerade für die Erhebung von Problemen von Vorteil.

5.3.1.4 *Psychologischer Test*

„Ein psychologischer Test [. . .] ist ein wissenschaftliches Datenerhebungsverfahren, das aus mehreren Testaufgaben (Testbogen/Testmaterial) sowie festgelegten Regeln zu deren Anwendung und Auswertung (Testmanual) besteht. Ziel eines psychologischen Tests ist es, ein latentes psychologisches Merkmal (Konstrukt) – typischerweise eine Fähigkeit oder Persönlichkeitseigenschaft – in seiner absoluten oder relativen Ausprägung zu Forschungszwecken oder für praktische Entscheidungen zu erfassen“ (Döring & Bortz, 2016, S.431).

Indikatoren:

„Ein psychologischer Test ist immer dann für Forschungszwecke besonders geeignet (Indikation) wenn zentrale und etablierte psychologische Merkmale (z. B. Intelligenz, Extraversion, Depression) mit einem besonders gut geprüften Verfahren erfasst werden sollen, Ergebnisse einer Studie mit Ergebnissen anderer Studien oder einer Referenzstichprobe vergleichbar sein sollen, leistungsbezogene Merkmale objektiv – d. h. unabhängig von subjektiven Selbsteinschätzungen der Testpersonen – erfasst werden sollen (z. B. Intelligenz, Berufseignung, Schuleignung)

(Döring & Bortz, 2016, S.431)“. Diese Art von Test ist nicht geeignet um Probleme Studierender zu erfassen, es sollen keine latenten psychologischen Merkmale erfasst werden, Vergleichbarkeiten und leistungsbezogene Merkmale sind nicht nötig. Die Selbsteinschätzungen der Testpersonen zur Erfassung ihrer Probleme sind relevant, jedoch sollen auch Fremdeinschätzungen vorgenommen werden, um mögliche verdeckte Probleme ebenfalls zu identifizieren.

5.3.1.5 *Physiologische Messung*

„Physiologische Messungen – Die physiologischen Messungen dienen der objektiven Erfassung und Quantifizierung bestimmter Merkmale physiologischer Prozesse in unterschiedlichen Organsystemen des Körpers mittels entsprechender Messgeräte. Die erhobenen Merkmale (z. B. Herzschlagfrequenz) werden als physiologische Indikatoren oder Biosignale bezeichnet. Meist werden mehrere Biosignale integriert erfasst und ausgewertet (z. B. Hirnaktivität und Blickbewegungen)“ (Döring & Bortz, 2016, S.501).

Indikatoren:

- „Physiologische Indikatoren lassen sich durch die Untersuchungspersonen nicht so direkt und gezielt beeinflussen bzw. verfälschen wie Selbstauskünfte, welche z. B. bei Bedarf sozial erwünscht gestaltet werden können“ (Döring & Bortz, 2016, S.502). Dieser Indikator erscheint für das Vorhaben zum Teil vorteilhaft, da die Probleme Studierender, die sie tatsächlich aufweisen, erfasst werden sollen und nicht ausschließlich die, die sie angeben zu haben.
- „Gedächtnisfehler sind eine häufige Verfälschungsquelle bei Selbstauskünften (z. B. die Häufigkeit des eigenen Verhaltens wird nicht korrekt erinnert), die bei physiologischen Messungen aufgrund der Objektivität der Methode keine Rolle spielen (z. B. automatische Aufzeichnung der Herzschlagfrequenz per EKG)“ (Döring & Bortz, 2016, S.502). Dieser Indikator wäre für das Vorhaben zum Teil vorteilhaft, da die Probleme Studierender, die sie tatsächlich haben, erfasst werden sollen und nicht ausschließlich die, die sie glauben zu haben.
- „Mit physiologischen Messmethoden lassen sich Daten über Phänomene erheben, die nicht bewusst wahrgenommen werden (Veränderung des Hautwiderstands durch Stress) und/oder nicht eindeutig verbalisiert werden können (genauer Anstieg der Herzschlagfrequenz)“ (Döring & Bortz, 2016, S.502). Dieser Indikator wäre für das Vorhaben zum Teil vorteilhaft, da die Probleme Studierender, die

sie tatsächlich haben, erfasst werden sollen und nicht ausschließlich die, die sie korrekt verbalisieren können.
- „Physiologische Messungen ermöglichen eine bis zu millisekundengenaue Erfassung von Daten im Zeitverlauf“ (Döring & Bortz, 2016, S.502). Dieser Indikator wäre für das Vorhaben zum Teil vorteilhaft, da die Probleme Studierender, die sie zu einem bestimmten Zeitpunkt im Modellierungsprozess haben, erfasst werden sollen.
- „Physiologische Daten erlauben einen Einblick in Teilprozesse der Informations- und Emotionsverarbeitung, den man durch Verhaltensmaße nicht bekommt, denn letztere sind immer nur das Endprodukt der Verarbeitung (Sommer, Ulrich, & Leuthold, 1996; Sternberg, 2001)“ (Döring & Bortz, 2016, S.502). Dieser Indikator spielt für das Vorhaben keine Rolle.

5.3.1.6 *Dokumentenanalyse*

Die wissenschaftliche Dokumentenanalyse definieren Döring und Bortz (2016) „die zielgerichtete, systematische und regelgeleitete Sammlung und Archivierung von vorhandenen (d. h. unabhängig vom Forschungsprozess produzierten) Dokumenten als Manifestationen menschlichen Erlebens und Verhaltens. Dabei kann es sich inhaltlich um persönliche oder offizielle Dokumente sowie formal um textuelle/verbal-schriftliche sowie um nicht-textuelle (visuelle, auditive, audiovisuelle, multimediale, hypermediale etc.) Dokumente handeln. An die Sammlung, Archivierung und Aufbereitung des Rohdatenmaterials schließt sich eine Auswertung der Dokumente hinsichtlich ihrer inhaltlichen und formalen Merkmale an. Bei der qualitativen Dokumentenanalyse stellt die interpretative Auswertung der Dokumente bereits die eigentliche qualitative Datenanalyse dar“ (Döring & Bortz, 2016, S.537).

Indikatoren:

- „Manche Forschungsthemen lassen sich mithilfe der Dokumentenanalyse besonders gut bearbeiten, weil die interessierenden Sachverhalte sich systematisch in Dokumenten niederschlagen: So greift z. B. die medien- und kommunikationswissenschaftliche Produkt- und Medieninhaltsforschung primär auf Dokumentenanalysen ausgewählter Mediendarstellungen zurück. [...] Auch wenn größere soziale Gebilde wie Institutionen im Fokus des Forschungsinteresses stehen, können Dokumente eine wichtige Datenquelle darstellen: z. B. schulische Curricula, Hausordnungen, Dienstvorschriften, Kranken-, Personal- und Gerichtsakten, Sitzungsprotokolle, Vereinssatzungen etc.“ (Döring & Bortz, 2016, S.537). Die zugrundeliegende

Fragestellung lässt sich nicht ausschließlich mit Hilfe dieses Indikators beantworten.

- „Die genuine Dokumentenanalyse zählt zu den non reaktiven Verfahren der Datenerhebung, da auf Dokumente zurückgegriffen wird, die unabhängig vom Forschungsprozess erzeugt wurden. Somit wird Rohdatenmaterial bearbeitet, das hinsichtlich Form und Inhalt nicht durch den Forschungsprozess beeinflusst ist“ (Döring & Bortz, 2016, S.537). Dieser Indikator trifft auf das Forschungsproblem, den Modellierungsprozess zu untersuchen, nicht zu. Die Untersuchung eines Prozesses von Studierenden lässt sich nicht anhand einer non reaktiven Erhebungsmethode bewerkstelligen.
- „Je nach Art der untersuchten Dokumente kann es sich bei der Dokumentenanalyse um eine sehr forschungsökonomische Form der Datenerhebung handeln. So können z. B. Informationen über ansonsten schwer zugängliche Minoritäten oder unterschiedliche Kulturen anhand von öffentlichen Dokumenten der Onlinekommunikation (z. B. Online-Diskussionsforen, Online- Profile; Online-Videos) in großer Menge und Vielfalt vom Schreibtisch aus gesammelt und ausgewertet werden“ (Döring & Bortz, 2016, S.537). Eine Studie wird sicherlich besser, je mehr Datenmaterial vorhanden ist. Wenn mehr Material effizient gewonnen werden kann, ist das sicherlich positiv zu bewerten.

5.3.1.7 *Zusammenfassung*

Ziel der Erhebung war die Identifikation von Problemen im Verlauf einer Modellierung. Auf Basis dieser Zielsetzung wurden Einschätzungen zur Wahl der Methoden getroffen. Es lässt sich keine eindeutige Methode zur Erhebung der Probleme Studierender im Modellierungsprozess ableiten. Deshalb wurde für das Gesamtvorhaben und der zugrundeliegenden Zielsetzung ein gemischter Methodenansatz, bestehend aus Fragebogen sowie Beobachtung mit, zusätzlich, anschließenden retrospektiven Interviews (bzw. eine Form der Think-Aloud Methode), gewählt. Fragebogen und Think-Aloud Methode wurden in Kombination eingesetzt, um sowohl Selbst- wie Fremdeinschätzungen zu erheben sowie um Studierendensicht und Dozierendensicht gleichermaßen zu berücksichtigen.

Im Folgenden werden nun Aufbau und erste Ergebnisse des Fragebogens beschrieben. Die Ergebnisse der quantitativ auswertbaren Items[22] des Fragebogens bzw. das zweite Item des Fragebogens sind als „Zwischenergebnis“ zu bewerten und fließen in den Auswertungsprozess der

22 Als Item werden Fragen oder Aufgaben in einem Fragebogen bezeichnet Döring und Bortz (2016).

Tabelle 5.4: Items der Online-Umfrage

#	Frage	Antwortmöglichkeit
1	Was fiel Ihnen während der Modellierung leicht?	Likert-Skala
1.1.	Mir ist eher die Notation des Diagramms leicht gefallen.	
1.2.	Mir ist eher der logische Aufbau des Diagramms leicht gefallen.	
2	Was fiel Ihnen schwer bei der Modellierung Ihres [Diagrammart]? An welcher Stelle hatten Sie Probleme?	offenes Textfeld
3	Was hätte Sie beim Lösen der Aufgabe bei der Modellierung Ihres [Diagrammart] unterstützt? (Auch unabhängig von eventuell aufgetretenen Problemen)	offenes Textfeld
4	Welche Fragen würden Sie zum Thema Modellierung von gerne noch stellen?	offenes Textfeld

Think-Aloud Studie ein. Anschließend wird das Design des zweiten Teils der Studie erläutert und der Prozess sowie Auswertung und Ergebnisse der Gesamtstudie vorgestellt.

5.3.2 *Online-Fragebogenstudie*

Die Online-Umfrage bestand aus vier Items, die vorrangig die Selbsteinschätzung der Studierenden zu aktuellen Problemen und zusätzlich Verbesserungsmöglichkeiten pro Diagrammtyp erheben sollten. In diesem Kapitel werden nur die ersten beiden Items betrachtet, Item 3 und 4 werden in den entsprechenden Kapiteln zu Hilfestellungen ausgewertet.

Das erste Item „Was fiel Ihnen während des Modellierung leicht?“ wurde getrennt in Logik und Notation abgefragt und war positiv formuliert, um Studierende nicht von vornherein negativ zu beeinflussen. Die Unterscheidung in Notation und Logik wurde gewählt, um eine Problem-Differenzierung innerhalb eines Diagramms zu gewährleisten. Durch die Wahl der Likert-Skala (anstatt eines Freitextfelds) von 1, trifft gar nicht zu bis 5, trifft voll zu, konnte eine quantitative Einschätzung der Studierenden pro Diagrammtyp ermittelt werden[23]. Item 2 erfragte in offener Form Probleme, die die Studierenden während der Modellierung des jeweiligen Diagrammtyps wahrnahmen.

Mit Hilfe von Item 1 lassen sich also quantitative Rückschlüsse hinsichtlich Problemen mit Notation und Logik in den Diagrammtypen ziehen, mit Hilfe von Item zwei wurde zusätzlich zur Literatur das initiale Codesystem für die Auswertung der Think-Aloud Studien erstellt.

Die Umfragen wurden pro Diagrammtyp gesondert erstellt und mittels Limesurvey[24] durchgeführt.

5.3.3 *Auswertung der Fragebogenstudie*

Die folgenden Abschnitte präsentieren die Ergebnisse der quantitativen Auswertung für Item 1 und die qualitative Auswertung für Item 2.

5.3.3.1 *Item 1*

Zunächst wird Item 1 betrachtet: Was fiel Ihnen während der Modellierung leicht?

ONEWAY DESKRIPTIVE STATISTIKEN FÜR ITEM 1.1. Für Item 1, bzw. 1.1. und 1.2. wurde deskriptive Statistik angewandt. Die numerische Zuordnung der Diagrammart ist wie folgt zu verstehen:

1) Zustandsdiagramm
2) Sequenzdiagramm
3) Klassendiagramm

23 Bei einer Likert Skala sind die Skalen-Items intervallskaliert und nicht nominal. Eine Likert Skala besteht also aus mehreren mindestens fünfstufigen Items, die zu einem Index durch Addition zusammengefasst werden. Es werden mehrere Items gebildet die nicht dichotom sind (Ja/Nein), sondern abgestuft (stimme voll zu - stimme gar nicht zu). Die Abstufungen können entweder mit Zahlen oder textuell belegt sein. Diese Antworten der einzelnen Items werden summiert. Durch die Betrachtung des Mittelwerts kann eine untersuchte Eigenschaft deutlich präziser abgebildet werden als durch ein einzelnes Item. Der Gesamtpunktwert für sich ist aussagekräftiger, weil die einzelnen Items schon intervallskaliert sind (Brosius, Haas & Koschel, 2016, S.49).

24 https://www.limesurvey.org/de/

4) Use-Case-Diagramm
5) Aktivitätsdiagramm

Für Item 1.1. „Mir ist eher die Verwendung der korrekten Notation leicht gefallen“, ergibt sich die Übersicht in Tabelle 5.5:

Tabelle 5.5: ONEWAY deskriptive Statistik für Item 1.1. In Spalte 1 wird die Diagrammart wie oben nummeriert verwendet. Das Item wurde mit einer fünfstufigen Skala konstruiert.

#	N	mean	med	SD	SE	95%-Konfidenzintervall	
						Untergrenze	Obergrenze
1	51	3.55	4	1.045	.146	3.26	3.84
2	52	**2.44**	2	1.037	.144	**2.15**	**2.73**
3	49	3.59	4	1.189	.170	3.25	3.93
4	75	3.31	3	1.026	.119	3.07	3.54
5	45	3.53	4	.991	.148	3.24	3.83
Ges.	272	3.28		1.130	.069	3.14	3.41

TESTS AUF NORMALVERTEILUNG ITEM 1.1 Sowohl der Kolmogorov-Smirnov Test (Tabelle 5.6) wie auch der Shapiro-Wilk Test (Tabelle 5.7) auf Normalverteilung zeigen für alle Diagrammtypen signifikante Werte, d. h. die Daten sind nicht normalverteilt.

Tabelle 5.6: Test auf Normalverteilung nach Kolmogorov-Smirnov für Item 1.1.

#		Kolmogorov-Smirnov		
		Statistik	df	Signifikanz
Notation	1	0.216	51	0.000
	2	0.242	52	0.000
	3	0.206	49	0.000
	4	0.244	75	0.000
	5	0.259	45	0.000

ONEWAY DESKRIPTIVE STATISTIKEN FÜR ITEM 1.2. Für Item 1.2. „Mir ist eher der logische Aufbau des Diagramms leicht gefallen“, ergibt sich die Übersicht in Tabelle 5.8.

Tabelle 5.7: Test auf Normalverteilung nach Shapiro-Wilk für Item 1.1.

#		Shapiro-Wilk Statistik	df	Signifikanz
Notation	1	0.898	51	0.000
	2	0.891	52	0.000
	3	0.870	49	0.000
	4	0.893	75	0.000
	5	0.864	45	0.000

Tabelle 5.8: ONEWAY deskriptive Statistik für Item 1.2.. Das Item wurde mit einer fünfstufigen Skala konstruiert.

#	N	mean	med	SD	SE	95%-Konfidenzintervall Untergrenze	Obergrenze
1	51	3.41	4	1.080	.151	3.11	3.72
2	52	**2.67**	3	1.232	.171	**2.33**	**3.02**
3	49	3.37	3	.951	.136	3.09	3.64
4	77	3.25	3	1.028	.117	3.01	3.48
5	44	3.45	4	1.044	.157	3.14	3.77
Ges.	273	3.22		1.097	.066	3.09	3.35

Tabelle 5.5 und Tabelle 5.8 zeigen, dass Studierende sowohl für 1.1. wie auch für Subitem 1.2. dieses Diagramm als am Schwierigsten einschätzen.

TESTS AUF NORMALVERTEILUNG ITEM 1.2. Sowohl der Kolmogorov-Smirnov Test (Tabelle 5.9) wie auch der Shapiro-Wilk Test (Tabelle 5.10) auf Normalverteilung zeigen für alle Diagrammtypen signifikante Werte, d. h. die Daten sind nicht normalverteilt.

LEVENETEST FÜR ITEM 1.1. Für Item 1.1 und 1.2. wurde nun weiterhin ein Levenetest durchgeführt.

Ho des Levenetests prüft auf Homogenität der Varianzen. Dies ist hier nicht der Fall, da der Levenetest (.388) nicht signifikant wird. Signifikanz wäre bei $p < .005$ erreicht. Die Varianzen sind also nicht homogen und damit nicht gleichverteilt.

Nach Bestätigung durch einen Leventest wurde eine Gleichverteilung der Varianzen ausgeschlossen und eine Varianzanalyse durchgeführt.

Tabelle 5.9: Test auf Normalverteilung nach Kolmogorov-Smirnov für Item 1.2.

Diagrammart		Kolmogorov-Smirnov		
		Statistik	df	Signifikanz
Logik	1	0.256	51	0.000
	2	0.186	52	0.000
	3	0.217	49	0.000
	4	0.223	77	0.000
	5	0.222	44	0.000

Tabelle 5.10: Test auf Normalverteilung nach Shapiro-Wilk für Item 1.2.

Diagrammart		Shapiro-Wilk		
		Statistik	df	Signifikanz
Logik	1	0.890	51	0.000
	2	0.883	52	0.000
	3	0.900	49	0.001
	4	0.900	77	0.000
	5	0.902	44	0.001

Tabelle 5.11: Levenetest für Item 1.1.

Levene-Statistik	df1	df2	Signifikanz
1.038	4	267	**.388**

Tabelle 5.12: Levenetest für Item 1.2.

Levene-Statistik	df1	df2	Signifikanz
1.997	4	268	**.095**

LEVENETEST FÜR ITEM 1.2. Der Levenetest für Item 1.2. ist ebenfalls nicht signifikant $p = .095$. Auch hier sind die Varianzen damit nicht homogen und es wird eine ANOVA (Döring & Bortz, 2016) durchgeführt. Die ANOVA wurde durchgeführt, da sie als robust gegenüber der Verteilung der Daten gilt, d. h. unabhängig davon, ob diese normalverteilt sind oder nicht (Wilcox, 2012).

Da beide Levenetests nicht signifikant sind, wurden für beide Items jeweils einfaktorielle ANOVAs durchführt. Mit Hilfe der einfaktoriellen ANOVA wird festgestellt, ob zwischen den Gruppen (hier zwischen den Diagrammarten) Unterschiede festzustellen sind. Wird diese signifikant ($p < .05$), werden Unterschiede festgestellt, anderenfalls nicht.

Tabelle 5.13: Einfaktorielle ANOVA für Item 1.1.

	Quadrat-summe	df	Mittel der Quadrate	F	Signifi-kanz
Zwischen den Diagrammarten	47.882	**4**	11.971	**10.710**	**.000**
Innerhalb den Diagrammarten	298.438	**267**	1.118		
Gesamt	346.32	271			

EINFAKTORIELLE ANOVA FÜR ITEM 1.1. Die Bewertung des Items 1.1. unterscheidet sich signifikant für die verschiedenen Diagramme $F(4,267) = 10.710, p < 0.001$.

Tabelle 5.14: Einfaktorielle ANOVA für Item 1.2.

	Quadrat-summe	df	Mittel der Quadrate	F	Signifi-kanz
Zwischen den Diagrammarten	20.966	**4**	5.242	**4.585**	**.001**
Innerhalb den Diagrammarten	306.404	**268**	1.143		
Gesamt	327.370	272			

EINFAKTORIELLE ANOVA FÜR ITEM 1.2 Die Bewertung des Items 1.2. unterscheidet sich signifikant für die verschiedenen Diagramme $F(4,268) = 4.585, p = 0.001$.

MEHRFACHVERGLEICHE FÜR ITEM 1.1. Abhängige Variable: Item 1.1

Tukey-HSD

Signifikante Unterschiede zeigen sich bei Diagrammart 2 (Sequenzdiagramm), $p < 0.001$ im Vergleich zu 1,3,4,5.

Tabelle 5.15: Mehrfachvergleiche für Item 1.1.: Die Differenz der Mittelwerte ist auf dem Niveau 0.05 signifikant.

#	#	Mittlere Differenz	SE	Signifikanz	95%-Konfidenzintervall Untergrenze	Obergrenze
1	2	1.107*	.208	**.000**	.53	1.68
	3	-.043	.211	1.000	-.62	.54
	4	.242	.192	.714	-.28	.77
	5	.016	.216	1.000	-.58	.61
2	1	-1.107*	.208	**.000**	-1.68	-.53
	3	-1.150*	.210	**.000**	-1.73	-.57
	4	-.864*	.191	**.000**	-1.39	-.34
	5	-1.091*	.215	**.000**	-1.68	-.50
3	1	.043	.211	1.000	-.54	.62
	2	1.150*	.210	**.000**	.57	1.73
	4	.285	.194	.584	-.25	.82
	5	.059	.218	.999	-.54	.66
4	1	-.242	.192	.714	-.77	.28
	2	.864*	.191	**.000**	.34	1.39
	3	-.285	.194	.584	-.82	.25
	5	-.227	.199	.787	-.77	.32
5	1	-.016	.216	1.000	-.61	.58
	2	1.091*	.215	**.000**	.50	1.68
	3	-.059	.218	.999	-.66	.54
	4	.227	.199	.787	-.32	.77

MEHRFACHVERGLEICHE FÜR ITEM 1.2. Abhängige Variable: Item 1.2.

Tukey-HSD

Signifikante Unterschiede zeigen sich wiederum bei Diagrammart 2 (Sequenzdiagramm), $p < 0.05$ im Vergleich zu 1,3,4,5.

KRUSKAL-WALLIS-TEST FÜR ITEM 1.1. Um die Resultate der ANOVA abzusichern, da die ANOVA zwar als robust gilt, aber die Normalverteilungsannahme verletzt ist, wurde für beide Subitems je ein

Tabelle 5.16: Mehrfachvergleiche für 1.2: Die Differenz der Mittelwerte ist auf dem Niveau 0.05 signifikant.

#	#	Mittlere Differenz	SE	Signifikanz	95%-Konfidenzintervall Untergrenze	Obergrenze
1	2	.739*	.211	**.005**	.16	1.32
	3	.044	.214	1.000	-.54	.63
	4	.165	.193	.913	-.37	.70
	5	-.043	.220	1.000	-.65	.56
2	1	-.739*	.211	**.005**	-1.32	-.16
	3	-.694*	.213	**.011**	-1.28	-.11
	4	-.574*	.192	**.025**	-1.10	-.05
	5	-.781*	.219	**.004**	-1.38	-.18
3	1	-.044	.214	1.000	-.63	.54
	2	.694*	.213	**.011**	.11	1.28
	4	.121	.195	.972	-.42	.66
	5	-.087	.222	.995	-.70	.52
4	1	-.165	.193	.913	-.70	.37
	2	.574*	.192	**.025**	.05	1.10
	3	-.121	.195	.972	-.66	.42
	5	-.208	.202	.842	-.76	.35
5	1	.043	.220	1.000	-.56	.65
	2	.781*	.219	**.004**	.18	1.38
	3	.087	.222	.995	-.52	.70
	4	.208	.202	.842	-.35	.76

Kruskal-Wallis-Test kalkuliert, da dieser auch bei nicht normal verteilten Datensätzen angewendet werden kann. Er kann auch anstelle der ANOVA verwendet werden (siehe z.B. Oelerich, 2005; Siegel, 1956, S.151; Du Prel, Röhrig, Hommel & Blettner, 2010, 12).

Auch hier ist der Unterschied zwischen den Gruppen signifikant und bestätigt somit das Ergebnis der ANOVA für Item 1.1..

KRUSKAL-WALLIS-TEST FÜR ITEM 1.2. Auch für Item 1.2. der den Schwierigkeitsgrad im Sinne des logischen Aufbaus von Diagrammen

Tabelle 5.17: Kruskal-Wallis-Test für Item 1.1.

Ränge			
	#	N	Mittlerer Rang
Notation	1	51	154.40
	2	52	81.75
	3	49	157.31
	4	75	138.63
	5	45	153.27
	Gesamt	272	

Tabelle 5.18: Statistik für Kruskal-Wallis-Test zu Diagrammarten in Item 1.1.

Statistik für Test	
	Notation
Kruskal-Wallis	35.639
df	4
Asymptotische Signifikanz	.000

Tabelle 5.19: Kruskal-Wallis-Test für Item 1.2.

Ränge			
	#	N	Mittlerer Rang
Logik	1	51	150.08
	3	49	145.05
	4	77	137.53
	5	44	151.78
	Gesamt	273	

abfragt ist der Kruskal-Wallis-Test signifikant und bestätigt das Ergebnis der ANOVA.

Die Varianzanalysen für Item 1.1., das die Notation eines Diagramms in den Vordergrund stellt und Item 1.2., das die Logik eines Diagramms beurteilen lässt sowie der jeweilige Kruskal-Wallis-Test, zeigen signifikante Unterschiede im Vergleich zu den anderen abgefragten Diagrammtypen. Daraus kann geschlossen werden, dass die befragten Studierenden

Tabelle 5.20: Statistik für Kruskal-Wallis-Test zu Diagrammarten in Item 1.2.

Statistik für Test	
	Logik
Kruskal-Wallis	13.978
df	4
Asymptotische Signifikanz	.007

aus den Diagrammtypen Use-Case-Diagrammen, Klassendiagrammen, Zustandsautomaten, Aktivitätsdiagrammen und Sequenzdiagrammen, Letzteres sowohl hinsichtlich Notation wie auch hinsichtlich Logik als am Schwierigsten beurteilen.

5.3.3.2 *Item 2: Initiales Codesystem*

In der folgenden Tabelle werden alle identifizierten Probleme, aus Item 2 und die mit Hilfe des SLRs identifizierten Probleme, dargestellt. Die zweite Spalte enthält die identifizierten Quellen und/oder die Angabe, ob dieses Problem auch von Studierenden in der Umfrage genannt wurde (dann ist dies mit dem „Umfrage“ gekennzeichnet). Beschreibungen der Probleme, werden in der Auswertung in Abschnitt 5.3.5 geliefert.

Tabelle 5.21: Initiales Codesystem für das Aktivitätsdiagramm

Code/Problem	Quelle
Probleme mit der Benutzung der Modellierungssoftware	Umfrage, (Stikkolorum et al., 2018)
Aufteilung von Verantwortlichkeiten/Swim-Lanes	Umfrage
Definition und Umfang Aktion und Aktivität	Umfrage
Verwendung von Activity-Final/Flow-Final	Umfrage
Festlegen von Ausnahmezuständen/-szenarien	Umfrage
Verwendung von Join/Fork	(Kruus et al., 2014)
Festlegen der Verbindungsart	Umfrage
Aufbau des Diagramms im Allgemeinen	Umfrage
Umfang des Diagramms	Umfrage
Verwechslung mit Zustandsdiagramm	Umfrage, (Siau & Loo, 2006)

Code/Problem	Quelle
Fork statt Entscheidungsknoten	(Kruus et al., 2014)
Darstellung optionaler und paralleler Schritte	Umfrage
Abläufe in Schleifen	Umfrage
Detailgrad der Aktivitäten/Aktionen	Umfrage
Kontrollfluss = zeitliche Reihenfolge?	Umfrage

Tabelle 5.22: Initiales Codesystem für das Klassendiagramm

Code/Problem	Quelle
Probleme mit der Benutzung der Modellierungssoftware	(Stikkolorum et al., 2018), Umfrage
Festlegen von Klassen	(Sien, 2011), (Or-Bach & Lavy, 2004) (Thomasson et al., 2006b), (Stikkolorum et al., 2018), Umfrage
Festlegen von Methoden	(Or-Bach & Lavy, 2004), (Stikkolorum et al., 2018)
Umgang mit Boundary/Control Klassen	(Sien, 2011)
Beziehungen zwischen Klassen	(Stikkolorum et al., 2018), Umfrage
Verwendung Aggregation/Komposition	Umfrage
Verwechslung zwischen Aggregation und Vererbung	(Kruus et al., 2014)
Aufbau des Diagramms im Allgemeinen	Umfrage
Korrekte Notation im Allgemeinen	Umfrage
Umfang des Diagramms	(Or-Bach & Lavy, 2004), Umfrage
Methoden, die keinen Klassen zugeordnet werden	(Bolloju & Leung, 2006)
Falsch zugeordnete Methoden	(Bolloju & Leung, 2006)
Falsche Multiplizitäten	(Bolloju & Leung, 2006)

Code/Problem	Quelle
Falsch zugeordnete Attribute	(Bolloju & Leung, 2006), (Thomasson et al., 2006b)
Falsche Verwendung von Generalisierungs-Spezialisierungshierarchien	(Bolloju & Leung, 2006)
Methoden können nicht realisiert werden mit derzeitigen Attributen und Beziehungen	(Bolloju & Leung, 2006)
Unzureichende Unterscheidung in Subklassen	(Bolloju & Leung, 2006)
Redundante Attribute	(Bolloju & Leung, 2006)
Isolierte Klassen	(Thomasson et al., 2006b)
Referenzen auf nicht existierende Klassen	(Thomasson et al., 2006b)
Hinzufügen irrelevanter Klassen	(Or-Bach & Lavy, 2004)
ungeeignete Klassen (modellierte Klassen könnten Attribute oder Methoden sein)	(Or-Bach & Lavy, 2004)
Modellierung prozeduraler, statt statischer Diagramme	(Or-Bach & Lavy, 2004)
Hinzufügen irrelevanter Attribute	(Or-Bach & Lavy, 2004)
Verwendung von Attributen statt Methoden	(Or-Bach & Lavy, 2004)
Abstraktion: Probleme mit implementierten Methoden in Attributen	(Or-Bach & Lavy, 2004)
Abstraktion: Problem abstrakte Methoden zu verwenden	(Or-Bach & Lavy, 2004)
Polymorphismus	(Or-Bach & Lavy, 2004), Umfrage
Identifikation geeigneter Generalisierung -Spezialisierungshierarchien	(Sien, 2011)
Identifizieren von Beziehungen zwischen Klassen aus dem Text	(Stikkolorum et al., 2018)
Unterscheidung zwischen Attribut und Klasse	(Stikkolorum et al., 2018)

Code/Problem	Quelle
Identifizieren von Attributen	(Stikkolorum et al., 2018)
Schwierigkeiten beim Verständnis von Einschränkungen in Klassendiagrammen	(Siau & Loo, 2006)
Detailgrad bei Vererbung	Umfrage
Verwendung von Multiplizitäten	Umfrage
Verwendung von Boundary und Control Klassen	Umfrage
Unterschied Assoziation und Komposition	Umfrage

Tabelle 5.23: Initiales Codesystem für das Sequenzdiagramm

Code/Problem	Quelle
Festlegen der interagierenden Klassen	(Stikkolorum et al., 2018), (Sien, 2011), Umfrage
Aufbau des Diagramms im Allgemeinen	(Stikkolorum et al., 2018), Umfrage
Korrekte Notation im Allgemeinen	Umfrage
Verwendung von Fragmenten	Umfrage
Verwendung synchroner Nachrichten	(Stikkolorum et al., 2018), (VanderMeer & Dutta, 2009), Umfrage
Verwendung asynchroner Nachrichten	(Stikkolorum et al., 2018), (VanderMeer & Dutta, 2009), Umfrage
Darstellung der Interaktion zwischen Objekten	(Stikkolorum et al., 2018),(Bolloju & Leung, 2006)
Umfang des Diagramms	(VanderMeer & Dutta, 2009), Umfrage
Verwendungszweck des Diagramms	Umfrage
Zerlegen unwichtiger Anforderungen in detaillierte Verantwortlichkeiten	(VanderMeer & Dutta, 2009)
Bestimmung der Verantwortlichkeiten	(VanderMeer & Dutta, 2009), Umfrage, (Sien, 2011)

Code/Problem	Quelle
Betrachtung des Systems als Blackbox	(Stikkolorum et al., 2018), Umfrage
Erstellen eines Top Level Diagramms, ohne Implementierungsdetails	(Stikkolorum et al., 2018)
Identifizieren von Sequenzen aus Anwendungsfall	(Stikkolorum et al., 2018), Umfrage
Verwendung von Antwortnachrichten	(Stikkolorum et al., 2018)
Identifizierung der zentralen Lebenslinie	(Stikkolorum et al., 2018)
Fehlende Verantwortlichkeiten	(Sien, 2011)
Fehlende Objekte	(Sien, 2011)
Fehlende Controller- Objekte	(Sien, 2011)
Ungeeignete Objekte (Objekte beziehen sich nicht auf die im Klassendiagramm identifizierten Klassen)	(Sien, 2011)
Schwierigkeiten die richtige zu sendende Nachricht zu identifizieren	(Sien, 2011), (Stikkolorum et al., 2018), Umfrage
Schwierigkeiten Zusammenhang zu Klassendiagramm herzustellen	(Sien, 2011), Umfrage
Fehlende Nachrichten	(Bolloju & Leung, 2006)
Fehlende Parameter	(Bolloju & Leung, 2006), (Sien, 2011)
Fehlende Objekte	(Bolloju & Leung, 2006)
Falsche Zuweisung von Verantwortlichkeiten	(Bolloju & Leung, 2006), (VanderMeer & Dutta, 2009)
Fehlender initialer Trigger	(Bolloju & Leung, 2006)
Return zu einem Objekt ist unterschiedlich zum Aufrufer	(Bolloju & Leung, 2006), Umfrage
Klasse/Objekt gehört nicht zum Klassendiagramm	(Bolloju & Leung, 2006)
Verwendung von Nachrichtenparametern, bevor Werte verfügbar sind	(Bolloju & Leung, 2006)
Zeitlich korrekte Abfolgen identifizieren	Umfrage

Tabelle 5.24: Initiales Codesystem für das Use-Case-Diagramm

Code/Problem	Quelle
Verwendung der korrekten Pfeilrichtung	(Kruus et al., 2014), Umfrage
Festlegen von Use-Cases	(Stikkolorum et al., 2018), (Kruus et al., 2014), (Bolloju & Leung, 2006), Umfrage
Festlegen von Akteuren	(Stikkolorum et al., 2018), Umfrage
Verwendung von Extend- und Include-Beziehungen	(Stikkolorum et al., 2018), (Bolloju & Leung, 2006), Umfrage
Festlegen der Beziehungen zwischen Use-Cases	(Kruus et al., 2014), (Stikkolorum et al., 2018), (Bolloju & Leung, 2006), Umfrage
Verwendung der korrekten Pfeilart	(Stikkolorum et al., 2018), Umfrage
Probleme mit der Benutzung der Modellierungssoftware	(Stikkolorum et al., 2018), Umfrage
Versuch der Modellierung einer Abfolge der Use-Cases	(Kruus et al., 2014)
Festlegen des Detailgrades des Diagramms	(Kruus et al., 2014), Umfrage
Korrekte Notation im Allgemeinen	(Bolloju & Leung, 2006)
Aufbau des Diagramms im Allgemeinen	(Stikkolorum et al., 2018)
Kommunikation im System	(Stikkolorum et al., 2018)
Einfache Methoden als Use-Cases	(Bolloju & Leung, 2006)
Zuweisen von Verantwortung zu den Konzepten, die gefunden wurden	(Stikkolorum et al., 2018)
Verwendung und Einsatz der Uses-Beziehung	(Siau & Loo, 2006)
Modellierung von Systemfunktionalität m. H. v. Use-Cases	(Kruus et al., 2014)
Benennung der Use-Cases	(Kruus et al., 2014)

Code/Problem	Quelle
Unfähigkeit, die Gesamtfunktionalität in separate Einheiten zu zerlegen	(Kruus et al., 2014), Umfrage
Generalisierung von Use-Cases	Umfrage
Generalisierung von Akteuren	Umfrage
Detailgrad eines Anwendungsfalls	Umfrage
Unterschied Extend- und Generalisierungs-Beziehung	Umfrage

Tabelle 5.25: Initiales Codesystem für das Zustandsdiagramm

Code/Problem	Quelle
Definition/Festlegen von Zuständen	(Stikkolorum et al., 2018), Umfrage
Verwendung von Events	(Stikkolorum et al., 2018), Umfrage
Verwendung von Zeitangaben	Umfrage
Verbindungen zwischen Zuständen (Transitionen) festlegen	(Stikkolorum et al., 2018), Umfrage
Aufbau des Diagramms im Allgemeinen	Umfrage
Korrekte Notation im Allgemeinen	(Stikkolorum et al., 2018)
Identifikation von Verhalten von Zuständen (aus dem Text)	(Stikkolorum et al., 2018), Umfrage
Anwendung von Guards (Wächtern)	(Stikkolorum et al., 2018), Umfrage
Richtige Benennung	(Stikkolorum et al., 2018), Umfrage
Umfang des Diagramms/Lebenszyklus	Umfrage
Abgrenzung zu Aktivitätsdiagramm	Umfrage, (Siau & Loo, 2006)

Tabelle 5.26: Initiales Codesystem für diagrammübergreifende Probleme

Code/Problem	Quelle
Verwechslung von Diagrammtypen	(Siau & Loo, 2006)
Probleme mit der Abstraktion von Diagrammen	(VanderMeer & Dutta, 2009)
Probleme mit der Struktur eines Diagramms	(Stikkolorum et al., 2018), Umfrage
Merken von allen Konzepten, Konstrukten, Techniken	(Stikkolorum et al., 2018)
Identifizieren von Elementen und deren Beziehungen zueinander	(Stikkolorum et al., 2018)
Umsetzen des mentalen Modells	(Duschl et al., 2014)
Probleme mit der Benutzung der Modellierungssoftware	Umfrage, (Stikkolorum et al., 2018)

5.3.4 *Think-Aloud-/Beobachtungsstudie*

Mit Hilfe des zweiten Teils der Studie, sollten Probleme Studierender direkt während des Modellierungsprozesses identifiziert werden, da dies bislang in den identifizierten Publikationen noch nicht untersucht wurde und man einerseits die Probleme besser verstehen wollte und andererseits das Auftreten weiterer Probleme erwartet wurde. Intendiert war zudem eine Reliabilitätssteigerung innerhalb der Gesamtdatenbasis: Für den Fall, dass in den Teilerhebungen ähnliche Ergebnisse produziert werden wird angenommen, dass die Ergebnisse in sich stimmig sind und damit ein höherer Reliabilitätsgrad erzielt wurde.

5.3.4.1 *Think-Aloud als Erhebungsmethode*

Bei der Methode des lauten Denkens („think-aloud method“, „think-aloud protocol“ [TAP], „think-aloud test“) werden Untersuchungsteilnehmende aufgefordert, all ihre Gedanken in Worte zu fassen, die während einer bestimmten Aktivität (z. B. einen Text übersetzen, eine Entscheidung treffen, ein Computerprogramm nutzen) auftreten. Die interessierende Aktivität zusammen mit dem lauten Denken wird in der Regel per Video aufgezeichnet. Laut Döring und Bortz (2016) gilt diese Methode als eine speziellen Form des Interviews. In dieser Arbeit wird diese Methode aufgrund ihres Aufbaus einer Mischform aus Beobachtung und Befra-

gung (Interview) zugeordnet (Frommann, 2005; Knorr & Schramm, 2012; Sandmann, 2014).

Die Erhebungsmethode Think-Aloud zielt darauf ab, Einsicht in die innere Gefühlswelt von Lernenden oder problemlösenden Personen zu gewähren. Der Begriff der inneren Gefühlswelt umfasst beispielhaft Gedankengänge oder Vorhaben des Lernenden. Die Methode des Lauten Denkens ermöglicht es in diesem Zusammenhang den kognitiven Verarbeitungsprozess zu beobachten, welcher zu einem mentalen Sinnbild des Lernenden führt (Konrad, 2010). Dedizierte Informationen aus dem Langzeitgedächtnis werden in das Kurzzeitgedächtnis Abb. 3.4 gerufen, um das vorliegende Problem zu lösen (Bilandzic & Trapp, 2000, S.187).

Konrad (2010) unterscheidet drei verschiedene Arten von Think-Aloud:

- Formen der Introspektion
- Formen der unmittelbaren Retrospektion
- Formen der verzögerten Retrospektion

Bei der Introspektion soll der Problemlösende während der Bearbeitung der Aufgaben seine Gedanken verbalisieren. Die Methode zielt somit auf die Verbalisierung der Inhalte des Kurzzeitgedächtnisses ab. Die Unmittelbare Retrospektion reiht sich zeitlich unmittelbar an die Introspektion an, bezieht sich im Unterschied zur Introspektion aber auf die innere Gefühlswelt, die noch nicht verbalisiert wurde. Die Form der verzögerten Retrospektion findet frühestens am Ende einer Aufgabenstellung statt oder kann auch erst Tage später vollzogen werden. Hierbei steht die Erläuterung von Denkprozessen und Gedankengänge während der Aufgabenbearbeitung im Vordergrund. Betont wird auch, dass diese Ansätze in der Praxis auch ineinander übergehen können. Es gibt viele verschiedene Begriffe, die alle im Zusammenhang mit dem Terminus bzw. der Methode Think-Aloud in Verbindung stehen. Dazu gehören z. B. Gedankenprotokolle, Think-Aloud Protocols, Talk-Aloud-Interviews oder Verbal Protocols (Verbalprotokolle). Diese Begriffe stellen alle Artefakte der Think-Aloud Methode dar und müssen somit gezielt dokumentiert, ausgewertet und interpretiert werden (Buber, 2007; Konrad, 2010; Sasaki, n. d., 4).

Das Konzept des Think-Aloud kann in allen Bereichen angewendet werden, in denen Problemlösestrategien oder innere Gedankenabläufe identifiziert werden sollen. Im Bereich der Lernforschung, genauer dem selbstregulierten Lernen scheint die Methode geeignet (Sandmann, 2014). Konrad (2010) nennt weiterhin den Bereich des Spracherwerbs, der Entscheidungsforschung sowie beispielsweise auch den Ansatz als Usability-Testmethode. Sandmann (2014) behauptet, dass die Forschungsmethode des Think-Aloud größtenteils eingesetzt wird, um individuelle Lern- und

Problemlösestrategien zu identifizieren. Dabei ist explizit zu erwähnen, dass sich die Methodik nicht nur auf Individuen anwenden lässt, sondern auch im Kleingruppenkontext gut verwendet werden kann. Dabei stehen Lernmaterialien während der Erhebung besonders im Vordergrund, da diese den Informationsgehalt der Protokolle des Think-Aloud erheblich mitbestimmen. Gute Lernmaterialien lösen kontinuierliche Denkprozesse bei Lernenden aus, sodass Ermutigungen zum Lauten Denken, von Seiten des Versuchsleiters nicht mehr nötig sind. Des Weiteren muss bei der Gestaltung der Lernmaterialien auf Struktur und Altersangemessenheit geachtet werden. Beispielaufgaben, Lerntexte oder Problemlöse-Aufgaben sind Beispiele für obligatorische Lernmaterialien. Übungsaufgaben und Instruktionen zum Lauten Denken sind für den Anfang hilfreich. Eine Lernsitzung, in der die Methode des Think-Aloud angewendet wird, sollte laut Sandmann (2014) zum Beispiel folgendermaßen aussehen:

„1. Einführung in die Lernsitzung
2. Erklärung des Ziels der Lernsitzung und der Lernaufgabe
3. Instruktion zum Think-Aloud
4. Übungsaufgaben zum Think-Aloud
5. Bearbeitung des Lernmaterials
6. Technische Datensicherung "(Sandmann, 2014, S.185).

Die Methode wird oft durch ein nachträgliches Interview ergänzt, um die Entstehung von Gedankenprozesse und Handlungen während der Problemlösung zu erfahren (Völzke, 2012, S.34f.) Falls das Interesse an der Erhebung weiterer Faktoren besteht, können z. B. Fragebögen zu unterschiedlichen Zeitpunkten ausgebeben werden. Sandmann (2014) schlägt hierfür einmal den Zeitpunkt vor der Verkündung der Anweisungen zum Lauten Denken vor, oder nach der Bearbeitung des Lernmaterials. Es gilt allgemein zu beachten, dass keine Ermüdungseffekte bei den Lernenden eintreten. Hierbei ist die Länge der Lernsitzungen entscheidend, diese sollten in der Regel die Zeit von 60 bis 90 Minuten nicht überschreiten, da die Konzentrationsfähigkeit der Lernenden sonst gefährdet sein könnte. Alle Daten werden im Verlauf der Bearbeitung oder direkt danach, abhängig von der Form des Lauten Denkens wahlweise als Audio oder Video aufgenommen (Sandmann, 2014).

5.3.4.2 *Studiendesign — Erhebungsverlauf und Methodische Grenzen*

Im Folgenden wird nun der konkrete Verlauf der Studie aufgezeigt und ebenfalls methodische Grenzen.

STUDIENVERLAUF Für die Erhebung wurde ein mehrstufiges Konzept designed: Kleingruppen, wurden während des regulären Übungsbetriebs

und der Bearbeitung ihrer Aufgabe gebeten, nach einer kurzen Vorstellung des Vorhabens, möglichst laut zu Denken und im Anschluss an die Fertigstellung eines Diagramms den bereits erläuterten kurzen Fragebogen zu bearbeiten. Zusätzlich wurden Einzelpersonen (außerhalb des regulären Übungsbetriebs) rekrutiert. Dort lässt sich der Studienablauf wie folgt beschreiben:

1. Einstieg: Vorlesen des einleitenden Textes; Unterschreiben der Einwilligungserklärung für den Datenschutz
2. Hauptphase: Modellierung eines Diagramms anhand einer vorgefertigen Aufgabenstellung, währenddessen freies Erzählen des Probanden; Beobachter hakt nach, wenn der Redefluss ins Stocken gerät.
3. Vertiefung: Beobachter fragen nach, wenn eine Handlung nicht nachvollziehbar war oder eine Aussage sehr allgemein erschien (Wirklich „immer“, „alle“, „nie“? Warum haben sie dort diese Pfeilart benutzt?)
4. Retrospektives Interview: Walkthrough durch das erstellte Diagramm
5. Abschluss: Verabschiedung des Probanden

Beide Vorhabensstränge sind der introspektiven Variante der Think-Aloud Methode zuzuordnen. Wie von Sandmann (2014) vorgeschlagen, wurde sowohl ein retrospektives Interview mit Einzelpersonen wie auch den Kleingruppen, nach jeder Think-Aloud Sitzung geführt.

Folgende Leitfragen, hier beispielhaft für das Use-Case-Diagramm, wurden dabei für die retrospektiven Interviews angewendet:

- Wie seid ihr vorgegangen?
- Wo hattet ihr Probleme? Gab es schon vorher Stolpersteine (schwierige Formulierungen im Text)?
- Wie habt ihr identifiziert, was was ist (also was Aktor, Use-Case, usw. ist)?
- Waren Dinge schon von vornherein klar? Schon beim Lesen?
- Ist es euch schwer gefallen, Use-Cases aus dem Text zu extrahieren? Warum, warum nicht?
- Wie erkennt ihr, dass zwischen zwei Anwendungsfällen (Use-Cases) eine Assoziation bestehen muss?
- Hattet ihr Probleme beim Unterscheiden oder der Anwendung von extend und include-Beziehungen?
- Wie ging es euch bei der Identifikation von Akteuren?
- Gab es Schwierigkeiten bei der Differenzierung zwischen einer Nutzerfunktion und einer Aktion der Software?

- Gibt es eine bestimmte Vorgehensweise wie ihr den Text nach geeigneten Funktionen analysiert?

Die Kleingruppen haben zusätzlich einen Fragebogen (Abschnitt 5.3.2) ausgefüllt, um weitere Aspekte, wie gewünschte Hilfsmittel und Probleme aus Selbstperspektive zu erfassen.

In zwei Sitzungen nahmen in einem Kurs fünf, bzw. vier Gruppen zu je zwei bis drei Studierenden teil, in einem weiteren Kurs nahmen in vier Sitzungen vier bis fünf Gruppen zu je zwei bis drei Studierenden teil, zusätzlich haben zwei Personen alleine gearbeitet. Die Teilnahme erfolgte freiwillig und unter Anonymisierung der Datensätze, zum Schutz der persönlichen Daten im Rahmen der Datenschutzgrundverordnung Europäisches Parlament und Rat der Europäischen Union (2016). Die Aufnahme des Vorgangs erfolgte über eine Videokamera, die schräg hinter dem Probanden/der Gruppe positioniert wurde. Den Studierenden in den beiden Einzelsitzungen standen jeweils Stift und Papier im Format DIN A3 zur Bearbeitung einer vorher festgelegten Modellierungsaufgabe zur Verfügung. Die Gruppen, die während des regulären Übungsbetriebs teilnahmen, verwendeten eine Modellierungssoftware ihrer Wahl und es wurde der Bildschirm der Gruppe gefilmt, um Aktionen bzw. Handlungen der Teilnehmer festzuhalten. Eine Aufnahme von vorne wurde ausgeschlossen, da angenommen wurde, dass alle Haupthandlungen in der Modellierungssoftware transparent werden. Zudem wurde angenommen, dass eine Aufnahme von hinten weniger störend auf die Teilnehmer wirkt.

METHODISCHE GRENZEN Wie die meisten Methoden zur Datenerhebung, unterliegt auch dieser Methode, laut Sandmann (2014), gewissen Grenzen in der Anwendung:

1. Es kann nicht ausgeschlossen werden, dass die gezielte Verbalisierung von kognitiven Prozessen zur Einschränkung dieser führt und somit zu einem geringeren Erfolg.
2. Es ist unklar, ob tatsächlich alle kognitiven Prozesse erfasst werden können, da diese zum Teil doch sehr unterbewusst ablaufen und somit schwer nach außen getragen werden können.
3. Die Rolle der sozialen Erwünschtheit spielt auch bei dieser Erhebungsform eine Rolle.
4. Es bleibt offen, in wie Fern die Fähigkeit des generellen Verbalisierens der Lernenden eine Rolle bei den Protokollen des Lauten Denkens spielen.

Nach der Erhebung liegen ca. 63 Stunden Video- und Tonmaterial zur Auswertung vor. Für dieses wird im Folgenden eine Auswertungsmethode ausgewählt.

5.3.4.3 *Wahl der Auswertungsmethode*

Für die Auswahl des Auswerteverfahrens wurde, wie bereits für das Studiendesign, Döring und Bortz (2016) als Standardwerk herangezogen, die einen Überblick über existierende Auswertungsmethoden geben. Diese werden im folgenden kurz definiert und bezogen auf die Anwendbarkeit der Daten bewertet:

- Die wissenschaftliche Dokumentenanalyse in qualitativer Form beschäftigt sich mit der Auswertung von bereits vorhandenen Dokumenten in textueller und nicht-textueller Form nach inhaltlichen und formalen Merkmalen. Diese Daten müssen außerhalb des Forschungsprozesses generiert worden sein (Döring & Bortz, 2016, S. 533), sie wurden also in Form und Inhalt nicht vom Forschungsprozess beeinflusst. Bei den erhobenen Daten handelt es sich um Ergebnisse aus Feldbeobachtungen und Antworten von offenen Fragen eines qualitativen Fragebogens. Beides wurde aufgrund der Forschungsfragen durchgeführt, damit gelten sie als forschungsgeneriert. Bereits vorhandene Daten aus der Literatur wurden lediglich zur Extraktion von bereits bekannten Problemen genutzt.
- Bei der narrativen Analyse werden autobiografische Erzählungen inhaltlich und formal ausgewertet. Der Fokus liegt dabei auf der Konstruktion der Identität der Erzählenden (Döring & Bortz, 2016, S. 600). Dies ist bei dieser Studie nicht von Interesse, da die Probleme der Studierenden unabhängig von ihrem jeweiligen Charakter extrahiert werden sollen.
- Die interpretative Phänomenologische Analyse legt ebenfalls den Fokus auf die Auswertung nach autobiografischen Aspekten, allerdings ist hierbei die Interpretation und Verarbeitung von einschneidenden Erlebnissen des Erzählenden von größter Bedeutung (Döring & Bortz, 2016, S. 600). Autobiografische Aspekte spielen für die Forschungsfrage keine Rolle.
- Die Konversationsanalyse dient der Analyse von Alltagsgesprächen und Gruppendiskussionen mit Fokus auf die Reaktion der interagierenden Personen miteinander in Form von non- oder paraverbaler Kommunikation (Döring & Bortz, 2016, S. 601). In unserer Feldbeobachtung steht vor allem der Inhalt des Gesprochenen, während der Gruppendiskussionen, im Fokus. Die non-verbale Kommunikation ist durch die fehlende Aufnahme von Körper und Gesicht nicht

möglich und auch grundsätzlich nicht von Interesse, daher ist diese Methode nicht geeignet.

- Die kritische Diskursanalyse extrahiert in der Analyse von Diskussionen unterschwellig und offensichtlich dargestellte Machtverhältnisse (Döring & Bortz, 2016, S. 602). Diese sozialen Autoritäten sind nicht von Interesse, weshalb diese Methode nicht für diese Forschungsarbeit geeignet ist.
- Die Tiefenhermeneutik beschäftigt sich mit der Auswertung von kulturellen Artefakten in Hinsicht auf „latente Bedeutungsinhalte" (Döring & Bortz, 2016, S. 602). Diese Aspekte werden in diesem Forschungsvorhaben nicht untersucht.
- Objektive Hermeneutik bezeichnet die Untersuchung von „latenten Sinnstrukturen und objektiven Bedeutungsstrukturen" (Döring & Bortz, 2016, S. 602), um psychische, soziale und kulturelle Aspekte über ihre Darstellungsart zu analysieren. Dabei steht ausschließlich die strukturelle Form, z. B. Satzbau und Menge an Adjektiven, im Mittelpunkt. Das Verständnis der Intention des Erzählers ist irrelevant (Döring & Bortz, 2016, S. 602). Psychische, soziale und kulturelle Aspekte stehen nicht im Fokus der Forschungsfrage. Deshalb ist diese Methode nicht geeignet.
- Die dokumentarische Methode beinhaltet als Ergebnis eine Typenbildung, basierend auf einer formulierenden und anschließend reflektierenden Interpretation, gefolgt von einer fallübergreifenden, komparativen Analyse. Diese Art der Auswertung ist u.a. geeignet für Gruppendiskussionen und Feldbeobachtungen, welche qualitativ ausgewertet werden (Döring & Bortz, 2016, S. 602f.). Für die Datengrundlage erscheint dieses Evaluationsverfahren nicht geeignet, es sollen keine Vergleichen zwischen Gruppen oder fallübergreifende Analysen getätigt werden.
- Die Grounded-Theory-Methodologie (GTM) beschreibt einen aufwändigen Prozess zur Datenauswertung, der durch die sogenannte „Triangulation", bzw. „Einkreisung" der jeweiligen Kategorien durch wiederholte Auswertung, zum Ergebnis führt. Sie beschränkt sich nicht auf die Methode der Evaluation, sondern legt den gesamten Forschungsprozess fest (Döring & Bortz, 2016, S. 603). Es finden sich Elemente der Grounded-Theory-Methodologie in der Qualitativen Inhaltsanalyse, in Hinsicht auf die kontinuierliche Anpassung der Kategorien, die sogenannten „Rückkoppelungsschleifen", während des Auswertungsprozesses. Abschließend ist die Grounded-Theory-Methodologie weder als Methode noch als Forschungsprozess geeignet. Die Studie ist in einen größeren Kontext eingebettet und unterliegt dem DSR. Zudem wurde vorab ein

Wissensbestand aus der Literatur erarbeitet, was als Vorwissen bzw. Voreingenommenheit des Forschers gewertet werden könnte, und in einigen Varianten der GTM nicht ohne Weiteres zulässig ist.
- Bei der qualitativen Inhaltsanalyse werden aus den gesammelten Daten Kategorien gebildet. Während der Auswertung kann das entwickelte Kategoriensystem gegebenenfalls erweitert werden (Döring & Bortz, 2016, S. 602f.). Diese Form der Auswertung eignet sich gut für den vorliegenden Anwendungsfall. Bereits existierende Daten aus der Literaturrecherche dienen als grundlegendes Kategoriensystem und werden durch die vorliegenden Daten manifestiert und erweitert.

Als Auswertungsverfahren wurde die qualitative Inhaltsanalyse gewählt, da zur Beantwortung der Forschungsfrage und bei den vorliegenden Daten eine inhaltliche Betrachtung (der Probleme der Studierenden während des Modellierungsprozesses) im Fokus steht und keine Immanenz, non verbale Kommunikation oder Dramaturgie betrachtet werden muss.

QUALITATIVE INHALTSANALYSE Die qualitative Inhaltsanalyse wird von einer Vielzahl methodenbeschreibender Literatur, allen voran Mayring (1991) definiert und in Varianten beschrieben. Mayring (2016) formuliert folgenden Grundgedanken dazu:

> „Qualitative Inhaltsanalyse will Texte systematisch analysieren, indem sie das Material schrittweise mit theoriegeleitet am Material entwickelten Kategoriensystem bearbeitet“ (Mayring, 2016, S.114).

Konkret unterschieden wird zwischen drei Varianten: Der Zusammenfassung, der Explikation und der Strukturierung. Das Ziel der zusammenfassenden Inhaltsanalyse besteht nach Mayring darin, das Textmaterial dergestalt auf einer vorgegebenen Abstraktionsebene zu reduzieren, dass die wesentlichen Inhalte erhalten bleiben. Der verbleibende Textkorpus soll ein überschaubares aber getreues Abbild des Grundmaterials darstellen (Mayring, 2016, S.115). Bei der Explikation besteht nach Mayring das Ziel der Analyse darin, bei einzelnen fraglichen Textteilen, seien es Begriffe oder Sätze, durch die kontrollierte Einbeziehung zusätzlichen Materials das Verständnis zu erweitern, indem es die jeweilige Textstelle erläutert, erklärt oder ausdeutet. Hierbei unterscheidet Mayring zwischen einer engen und weiten Kontextanalyse (Mayring, 2016, S.213). Bei der Strukturierung besteht das Analyseziel darin, „bestimmte Aspekte aus

dem Material herauszufiltern, unter vorher festgelegten Ordnungskriterien einen Querschnitt durch das Material zu legen oder das Material aufgrund bestimmter Kriterien einzuschätzen“ (Mayring, 1991, S.115).

Betrachtet man zunächst die drei Formen der Inhaltsanalyse im Hinblick auf die Fragestellung, entfallen die Formen der zusammenfassenden und die der explizierenden Inhaltsanalyse, da weder Material abstrahiert werden soll, noch durch Hinzufügen weiteren Materials weiter expliziert werden soll. Die strukturierende Inhaltsanalyse scheint die geeignete Variante zu sein, da bestimmte Inhaltsbereiche, nämlich die konkreten Probleme Studierender im Modellierungsprozess, extrahiert werden sollen und zwar auf Basis der bereits aus der Literatur und der Fragebogenstudie identifizierten Probleme und Fehler. Mayring (2016) betont bei der Wahl der strukturierende Inhaltsanalyse, dass diese verschiedene Ziele haben kann und unterscheidet hier zwischen einer formalen, inhaltlichen, typisierenden und skalierenden Strukturierung des Textmaterials:

- „Eine formale Strukturierung will die innere Struktur des Materials nach bestimmten formalen Strukturierungsgesichtspunkten herausfiltern“ (Mayring, 2016, S.99). Diese Art der Inhaltsanalyse wird nicht gewählt, da keine formalen Gesichtspunkte zu filtern sind.
- „Eine inhaltliche Strukturierung will Material zu bestimmten Themen, zu bestimmten Inhaltsbereichen extrahieren und zusammenfassen“ (Mayring, 2016, S.99). Diese Form der strukturierenden Inhaltsanalyse scheint geeignet, um die Probleme der Studierenden während der Modellierungsphase eines Szenarios zu identifizieren.
- „Eine typisierende Strukturierung will auf einer Typisierungsdimension einzelne markante Ausprägungen im Material finden und diese genauer beschreiben“ (Mayring, 2016, S.99). Diese Form der strukturierenden Inhaltsanalyse erscheint ebenfalls sinnvoll, um die Probleme der Studierenden während der Modellierungsphase eines Szenarios zu identifizieren.
- „Eine skalierende Strukturierung will zu einzelnen Dimensionen Ausprägungen in Form von Skalenpunkten definieren und das Material daraufhin einschätzen“ (Mayring, 2016, S.99). Die Schwierigkeiten der Studierenden sollen nicht bewertet oder skaliert werden, sondern zunächst nur identifiziert werden, weshalb diese Form der Strukturierung nicht in Frage kommt.

Nach obiger Bewertung können demnach die inhaltliche Strukturierung und die typisierende Strukturierung zur Auswertung der Daten angewendet werden. Da Mayring (2016) in seiner Beschreibung der Varianten empfiehlt, Typisierungen nur anzuwenden, wenn andere Analyseformen nicht in Frage kommen, da diese immer die Gefahren der

Verallgemeinerung und Verzerrungen bergen würden, wird die inhaltlich strukturierende Inhaltsanalyse gewählt (Mayring, 2016, S.106).

5.3.4.4 *Die inhaltlich strukturierende Inhaltsanalyse als Auswertungsmethode*

Im Folgenden soll der Ablauf einer strukturierenden qualitativen Inhaltsanalyse kurz erläutert werden (siehe Abb. 5.4).

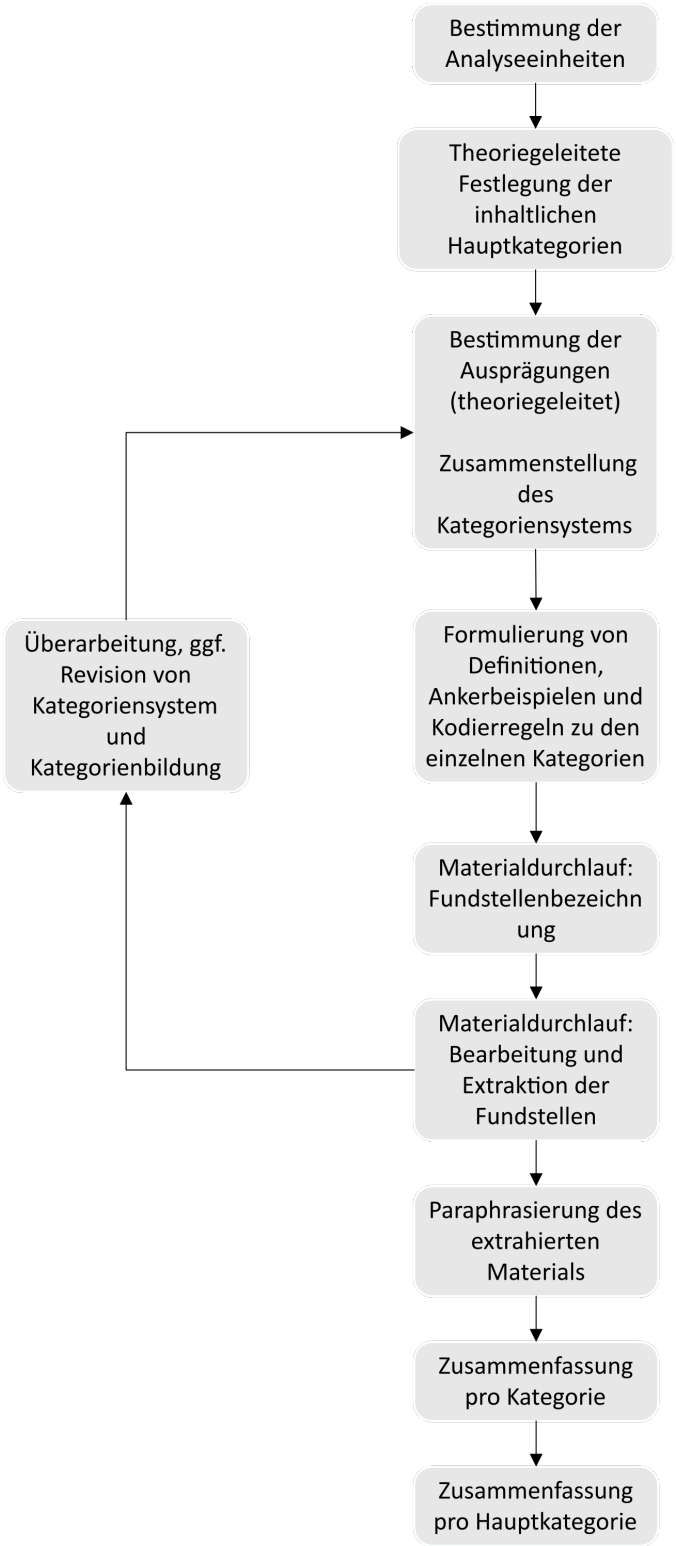


Abbildung 5.4: Ablaufmodell qualitativ strukturierender Inhaltsanalyse nach Mayring (2016)

Zunächst werden die entsprechenden Analyseeinheiten definiert. Dabei unterscheidet Mayring zwischen Kodiereinheit, Kontexteinheit und Auswertungseinheit:

- Kodiereinheit: Die Kodiereinheit bestimmt den kleinsten auswertbaren Materialbestandteil (Mayring, 2016, S.61). In der auszuwertenden Studie wird diese auf mehrere in Bezug auf die Kategorien bedeutende Worte festgelegt.
- Kontexteinheit: Definiert den größten Textbestandteil für eine Kategorie (Mayring, 2016, S.61). Dieses Element wird hier festgelegt auf den Bereich, der zum Verständnis des jeweiligen identifizierten Problems der Studierenden beiträgt.
- Auswertungseinheit: definiert, welche Textteile nacheinander ausgewertet werden(Mayring, 2016, S.61). Eine Auswertungseinheit entspricht (in dieser Arbeit) einer Audio- oder Videodatei, die sequentiell ausgewertet wird.

Im nächsten Schritt gilt es, die Hauptkategorien theoriegeleitet festzulegen. In der hier gewählten Form der strukturierenden Inhaltsanalyse geschieht die Kategorienbildung deduktiv, d. h. dass das Kategoriensystem, wie oben bereits beschrieben, vorab entwickelt wird (Mayring, 2016, S.68). Dennoch ist dieser Schritt nicht umfangreich in der Literatur beschrieben (Mayring, 2016, S.51; Krippendorff, 1980, S.76). Betrachtet man das Ablaufmodell (Abb. 5.4), ist auch eine induktive Kategorienentwicklung weiterhin möglich und möglicherweise auch nötig, wenn es beispielsweise einer Präzisierung, Modifizierung oder Differenzierung bedarf. Im vorliegenden Datensatz wurden als Hauptkategorien deshalb die in den Kursen behandelten Diagrammarten Use-Case-Diagramm, Klassendiagramm, Aktivitätsdiagramm, Zustandsautomat und Sequenzdiagramm gewählt. Für jede Subkategorie, die bereits identifizierten diagrammspezifischen Probleme aus der Literatur, wurde im Rahmen eines Memos eine Beschreibung der Kategorie angelegt und ein Beispiel angebracht. Schritt vier beinhaltet die Definition von Formalia, d. h. beispielsweise das Aufstellen von Kodierregeln und Beispielen und das Aufstellen von Definitionen. Diese Regeln werden nach dem ersten Entwurf pilothaft getestet und angepasst. Beispielsweise wird als Problem entweder eine Diskussion, eine zu hörende Unsicherheit, Fragen oder Fehler codiert.

Im Anschluss erfolgt die Materialauswertung und Fundstellen werden erfasst. Im vorliegenden Fall wurde diese Erfassung mit Hilfe von MAXQDA durchgeführt. Eine Transkription der Audio- und Videodaten erfolgte nicht. Gründe hierfür liegen in der Menge des Materials und der gegebenen Funktionalität von MAXQDA, Fundstellen direkt zu codieren.

Im Rahmen der Fundstellenanalysen wurden auch in der vorliegenden Auswertung Subkategorien und deren Definitionen induktiv entwickelt und in das Kategoriensystem aufgenommen.

Anschließend müssen die Fundstellen bearbeitet und extrahiert werden. Die folgenden Phasen stellen die Aufbereitung der Ergebnisse dar, die abhängig von der entsprechenden Zielsetzung bzw. Fragestellung angefertigt wird (Mayring, 2016, S.62). In Bezug auf die Fragestellung, welche diagrammspezifischen Probleme oder Schwierigkeiten die Studierenden während der Modellierung haben, erfolgt die Ergebnisdarstellung in tabellarischer Form, sodass letztendlich das identifizierte Problem und eine Beschreibung dessen erfasst wird (siehe Abschnitt 5.3.5).

Abschließend sollen nach dem allgemeinen inhaltsanalytischen Ablaufmodell Gütekriterien angewendet werden (Mayring (2016), S.62; siehe Abschnitt 5.3.5.1).

5.3.5 *Katalog existierender Probleme Studierender bei der Modellierung mit der UML und Zuordnung der Lernhindernisdimensionen*

Der folgende Abschnitt beschreibt die Ergebnisse der Auswertung. Die Tabellen zeigen den Problemkatalog differenziert nach Diagrammart. Durch das Symbol „>“ soll angezeigt werden, dass hier ein „Subcode“ (e.g. untergeordnetes Problem) genannt ist und das letzte zuvor genannte Problem ohne „>“ das übergeordnete Problem darstellt. Die Spalte Quelle enthält entweder die zugehörige Literaturangabe, die das Problem ebenfalls identifiziert oder die Angabe „Umfrage“, wenn ein Problem von Studierenden ebenfalls in der Fragebogenstudie genannt wurde. Keine Angabe in der dritten Spalte entspricht einem Finding, das ausschließlich in der Think-Aloud Studie identifiziert wurde.

An dieser Stelle sei darauf hingewiesen, dass dieser Katalog keinem Anspruch auf Vollständigkeit genügt und auch nicht in jedem Kontext Korrektheitskriterien beansprucht. Vielmehr dient der Katalog als Grundlage zu einer ersten Detektion von Problemen, die während der Modellierung von Diagrammen auftreten können. Ein hier gelistetes Problem muss beispielsweise nicht zwingend gegen das UML Metamodell verstoßen, sondern entspricht Richtlinien, die im Rahmen von Best Practices bekannt sind. Richtlinien sind meist subjektiv und nicht in jeder Hinsicht objektiv, sodass diese durchaus in verschiedenen Umständen unterschiedlich bewertet werden können.

Tabelle 5.27: Die Tabelle beschreibt identifizierte Probleme, die nicht spezifisch bei der Modellierung einer Diagrammart aufgetreten sind.

Code	Problembeschreibung	Quelle
Entscheidungen zur Relevanz von Informationen	Probleme zwischen relevanten und irrelevanten Informationen zu differenzieren, z. B. Hinzufügen irrelevanter Klassen oder Attribute	(Or-Bach & Lavy, 2004; Bolloju & Leung, 2006)
Umfang des Diagramms	Schwierigkeiten zu entscheiden, wann ein Diagramm vollständig ist.	(VanderMeer & Dutta, 2009)
Quantifizierung	Studierende treffen Entscheidungen aufgrund von Quantifizierungen, z. B. werden Rückfragen gestellt, wie viele Klassen erstellt werden sollen.	
Überblick behalten	Mangelnder Überblick aufgrund unstrukturierter Anordnung von Elementen	
Software	Probleme mit der Modellierungssoftware, z. B. Hinzufügen von Methoden und Attributen im Klassendiagramm.	(Stikkolorum et al., 2018)
Detaillierungsgrad	Probleme beim Bestimmen eines angemessenen Detailgrades, z. B. zu detailliert im Use-Case-Diagramm	(Kruus et al., 2014)
Probleme mit der Abstraktion von Diagrammen	Probleme bei steigender Abstraktion (von niedriger zu höherer Granularitätsebene) oder bei der Wahl eines adäquaten Abstraktionsniveaus	(Stikkolorum et al., 2018; VanderMeer & Dutta, 2009)
Verwechslung von Diagrammtypen	Es besteht beispielweise Unklarheit darüber, wann Zustandsdiagramm und wann Aktivitätsdiagramm verwendet wird	(Siau & Loo, 2006)
Struktur	Probleme mit der Struktur eines Diagramms	(Stikkolorum et al., 2018)

Code	Problembeschreibung	Quelle
Konstrukte, Techniken, Verwendung von Konzepten	Probleme bei der Verwendung von Konzepten, Konstrukten und der Notation	(Stikkolorum et al., 2018)
Identifizieren von Elementen und deren Beziehungen zueinander	Probleme beim Identifizieren von Elementen, wie z. B. dem Verhalten von Objekten (Zuständen und Events) aus dem Text	(Stikkolorum et al., 2018)
Umsetzen des mentalen Modells	Probleme beim Umsetzen des mentalen Modells in ein korrektes UML Modell, z. B. bei Klassen- und Zustandsdiagramm	(Duschl et al., 2014)
Zusammenhänge zwischen Diagrammen	Studierende haben Schwierigkeiten zu begreifen, dass zwischen einzelnen Diagramm typen auch Zusammenhänge bestehen (können).	

Tabelle 5.28: Identifizierte Probleme im Use-Case-Diagramm

Code	Problembeschreibung	Quelle
Verwechslung von Use-Cases mit Zuständen	Use-Cases werden behandelt, als wären sie Zustände.	
Zeitliche Abfolge von Use-Cases	Use-Cases werden in zeitlichen Abfolgen angeordnet.	(Kruus et al., 2014)
Verständnis-problem: Definition System	Es werden Use-Cases erstellt, die nicht Teil des Systems sind, oder es werden Use-Cases außerhalb der Systemgrenzen erstellt.	(Kruus et al., 2014)
Verwendungs-zweck des Use-Case-Diagramms	Studierende wissen nicht, wie ein Use-Case definiert ist.	

Code	Problembeschreibung	Quelle
Modellierung von Bedingungen zwischen Use-Cases	Studierende versuchen Bedingungen zwischen Informationen im Use-Case-Diagramm abzubilden.	(Bolloju & Leung, 2006)
Verwechslung mit Klassendiagramm	Use-Cases werden wie Klassen verwendet.	
Generalisierung von Use-Cases	Es ist nicht klar, wann die Generalisierungsbeziehung angewendet wird bzw. die Generalisierungs- und die Extend-Beziehung werden verwechselt.	Umfrage
> Generalisierung von Use-Cases: Verwendung	Es ist nicht klar, ob die Generalisierung von Use-Cases grundsätzlich möglich ist bzw. ob zwischen Use-Cases eine Generalisierung verwendet werden darf.	
Mehrere Generalisierungen eines Use-Cases	Es werden Use-Cases mittels Vererbung-Generalisierungsbeziehung miteinander verkettet.	
Pfeilrichtung der Extend-Beziehung	Die Pfeilrichtung einer Extend-Beziehung wird verkehrt herum verwendet.	(Kruus et al., 2014), Umfrage
Einsatz der Extend-Beziehung	Es ist unklar, ob zwischen zwei Elementen eine Extend-Beziehung verwendet werden darf.	(Bolloju & Leung, 2006; Stikkolorum et al., 2018), Umfrage
Verwendung der Extend-Beziehung	Es ist nicht klar, wann und wie die Extend-Beziehung modelliert wird.	(Bolloju & Leung, 2006; Stikkolorum et al., 2018), Umfrage
Pfeilrichtung der Include-Beziehung	Die Pfeilrichtung einer Include-Beziehung wird verkehrt herum verwendet.	(Kruus et al., 2014), Umfrage

Code	Problembeschreibung	Quelle
Verwendung der Include-Beziehung	Es ist nicht klar, wann und wie die Include-Beziehung modelliert wird.	(Bolloju & Leung, 2006; Stikkolorum et al., 2018), Umfrage
Einsatz der Include-Beziehung	Es ist unklar, ob zwischen zwei Elementen eine Include-Beziehung verwendet werden darf.	(Bolloju & Leung, 2006; Stikkolorum et al., 2018), Umfrage
Include-Beziehung/ Extend-Beziehung	Die Extend- und Include-Beziehung wird verwechselt.	(Stikkolorum et al., 2018), Umfrage
Festlegen von Akteuren	Es ist nicht klar, wie Akteure bestimmt werden.	(Stikkolorum et al., 2018)
Festlegen von Use-Cases	Die Differenzierung zwischen relevanten und irrelevanten Informationen fällt schwer.	(Bolloju & Leung, 2006; Kruus et al., 2014; Stikkolorum et al., 2018)
Ab wann ist ein Use-Case ein Use-Case	Es ist nicht klar, „wie viele Informationen“ für die Erstellung eines Use-Cases benötigt werden.	
Generalisierung (Pfeilart) von Akteuren	Die Bedeutung einer Vererbungsbeziehung zwischen Akteuren ist nicht klar.	Umfrage
Festlegen von sekundären Akteuren	Es ist nicht klar, wie sekundäre Akteure bestimmt werden.	
Beziehung Use-Case - Use-Case	Es ist nicht klar, in welcher Beziehung zwei Use-Cases miteinander stehen.	(Bolloju & Leung, 2006; Kruus et al., 2014; Stikkolorum et al., 2018), Umfrage

Code	Problembeschreibung	Quelle
Beziehung Akteur - Akteur	Es ist nicht klar, in welcher Beziehung zwei Akteure miteinander stehen.	
Beziehung Akteur - Use-Case	Es ist nicht klar, welche Beziehungstypen ein Akteur und ein Use-Case haben können.	(Kruus et al., 2014)
Benennung von Use-Cases	Der Use-Case verstößt gegen eine Richtline zur guten Benennung, es ist nur Substantiv statt Substantiv und Verb im Titel enthalten.	(Kruus et al., 2014)
Assoziation notwendig	Es fehlen Assoziationen zwischen Elementen.	
Zwei Use-Cases statt Einem	Die Use-Cases sind redundant. (Es gibt mehrere Use-Cases statt einem.)	
Zu viel Information für einen Use-Case	Der Use-Case enthält mehrere Use-Cases und wird nicht unterteilt.	
Rolle des Ausnahmeszenarios	Es ist nicht klar, wie sich das Ausnahmeszenario in textueller Form von der Extend-Beziehung unterscheiden.	

Tabelle 5.29: Identifizierte Probleme im Klassendiagramm

Code	Problembeschreibung	Quelle
Verwechslung mit Zustandsautomat	Klassen werden behandelt als wären sie Zustände.	
Verwechslung mit Sequenzdiagramm	Verwechslung der Assoziationstypen des Sequenzdiagramms (Nachrichten) mit den Assoziationstypen des Klassendiagramms.	

Code	Problembeschreibung	Quelle
Zweck des Diagramms	Studierende kennen den Verwendungszweck des Diagramms nicht, wann und wozu es eingesetzt wird.	
Aufbau des Diagramms im Allgemeinen	Elemente und Aufbau eines Klassendiagramms sind unklar.	Umfrage
Festlegen von Methoden		(Stikkolorum et al., 2018)
> Festlegen von Methoden: Methode oder Klasse	Es ist unklar, welche Information als Methode modelliert wird und welche als Klasse.	
> Festlegen von Methoden: Zuordnung von Methoden in Klassen	Es ist unklar, wie entschieden werden kann, welche Methode in einer Klasse implementiert wird.	(Bolloju & Leung, 2006)
Festlegen von Attributen		(Sien, 2011; Stikkolorum et al., 2018)
> Festlegen von Attributen: Benennung	Die sinnvolle Benennung von Attributen fällt schwer.	
> Festlegen von Attributen: Redundante Attribute	Es werden Attribute redundant zu anderen Attributen zugewiesen.	(Bolloju & Leung, 2006)
> Festlegen von Attributen: Konkrete Attributwerte	Attributen werden konkrete Werte zugewiesen.	(Bolloju & Leung, 2006)
Festlegen von Klassen		(Or-Bach & Lavy, 2004; Sien, 2011; Sien et al., 2010; Stikkolorum et al., 2018), Umfrage

Code	Problembeschreibung	Quelle
> Festlegen von Klassen: Modellierung nicht-funktionaler Requirements	Es werden nicht funktionale Anforderungen im Klassendiagramm modelliert.	
> Festlegen von Klassen: Sich häufig wiederholende Worte sind Klassen	Es wird angenommen, dass sich häufig wiederholende Worte Klassen sind.	
> Festlegen der Klassen auf Basis des Use-Case-Diagramms	Klassen werden ausschließlich auf Basis von zuvor definierten Use-Cases festgelegt.	
> Festlegen von Klassen: Klasse oder Attribut	Es ist nicht klar, wie festgelegt wird, welche Information als Klasse modelliert wird und welche als Attribut.	(Stikkolorum et al., 2018)
Umgang mit Boundary und Control Klassen	Es ist unklar, was eine Boundary Klasse ist.	(Sien, 2011), Umfrage
Vererbungs- bzw. Generalisierungs-prinzip	Die Modellierung des Vererbungsprinzips fällt schwer.	(Bolloju & Leung, 2006; Sien, 2011; Sien et al., 2010)
Unterschied zwischen gerichtete Assoziation und uses-Beziehung	Der Bedeutungsunterschied zwischen der gerichteten Assoziation und einer uses-Beziehung ist unklar.	
Unterschied zwischen Vererbungsbeziehung und Realisierung-beziehung (Pfeilart)	Der Bedeutungsunterschied zwischen Vererbung und Realisierung ist unklar.	

Code	Problembeschreibung	Quelle
Verwendung Komposition bzw. Aggregation	Die Pfeilarten Komposition und Aggregation werden verwechselt.	Umfrage
Verwechslung zwischen Aggregation und Vererbung	Die Pfeilarten Vererbung und Aggregation werden verwechselt.	(Kruus et al., 2014)
Alleinstehende Klassen	Es werden alleinstehende Klassen ohne Verbindungen modelliert.	(Thomasson et al., 2006b)
Beziehung zwischen Klassen:		(Stikkolorum et al., 2018), Umfrage
> Beziehung zwischen Klassen: Festlegen von Multiplizitäten	Es ist unklar, wie Multiplizitäten bestimmt werden.	(Bolloju & Leung, 2006)
> Beziehung zwischen Klassen: Bedeutung von Multiplizitäten	Die Bedeutung der Multiplizitäten ist unklar.	Umfrage
> Beziehung zwischen Klassen: Pfeilrichtung	Es ist unklar, ob die Pfeilrichtung relevant ist und wie die Pfeilrichtung bestimmt wird.	
> Beziehung zwischen Klassen: Beziehung über Attribut	Es ist unklar, wann ein Attribut modelliert werden muss und wann eine Assoziation genügt.	(Bolloju & Leung, 2006; Thomasson et al., 2006b)
Umfang des Diagramms	Studierende können nicht einschätzen, wie viel Information in das Diagramm mit aufgenommen werden muss und wie viel irrelevant ist.	(Or-Bach & Lavy, 2004), Umfrage
Detailgrad des Diagramms	Der Grad der Ausführlichkeit von Methoden und Attributen ist unklar.	

Code	Problembeschreibung	Quelle
Überblick behalten	Eine unstrukturierte Anordnung der Klassen sorgt für fehlenden Überblick.	

Tabelle 5.30: Identifizierte Probleme im Aktivitätsdiagramm

Code	Problembeschreibung	Quelle
Ablauflogik	Es gibt Probleme beim Festlegen der nacheinander ablaufenden Einzelschritte.	
Festlegen einer zwanghafte Reihenfolge	Es wird eine Reihenfolge bestimmt, wo keine bestimmt werden müsste.	
Zweck des Diagramms	Studierende verstehen den Zweck des Diagramms nicht.	
Unterschied Aktivitätsdiagramm und Zustandsdiagramm	Systemabläufe und die Einnahme von Zuständen eines Objekts in einem Zustandsdiagramm werden nicht unterschieden.	Umfrage
Unterschied Aktivitätsdiagramm und Sequenzdiagramm	Der Unterschied zwischen Aktivitäts- und Sequenzdiagramm ist nicht klar (Betrachtung des Systems als Blackbox).	
Verwechslung Aktion und Aktivität	Die Elemente Aktion und Aktivität werden verwechselt.	Umfrage
Verwendung von Return	Es ist nicht klar, was „return“ bedeutet und wie es verwendet wird.	
Endzustand aus verschiedenen Fällen heraus	Es ist nicht klar, wie man Vorgehen muss, wenn man aus mehreren Fällen zum Endzustand gelangen möchte.	
Mehrere Endzustände möglich?	Es ist nicht klar, ob es mehrere Endzustände geben kann.	

Code	Problembeschreibung	Quelle
Elemente des Aktivitätsdiagramms und Bedeutung	Die Elemente, die es in einem Aktivitätsdiagramm gibt sind nicht bekannt.	
Loop	Der Einsatz und das Ende von Loops ist nicht klar.	
Verwendung von Join und Fork	Die Bedeutungen der Elemente Fork und Join sind unklar.	(Kruus et al., 2014)
Verwendung von Activity-Final und Flow-Final	Der Unterschied zwischen Activity-Final und Flow-Final Verbindungen ist unklar.	Umfrage
Swim-Lanes	Das Element Swim-Lane ist nicht klar.	Umfrage
Pfeilbenennung	Eine geeignete Benennung der Pfeile fällt schwer.	
Umfang des Diagramms	Es ist nicht klar, wie viel Information das Diagramm enthalten muss.	Umfrage
Detailgrad des Diagramms	Das Diagramm enthält zu viele Informationen.	
Festlegen Ausnahmezustände und -szenarien	Es werden Ausnahmeszenarien im Aktivitätsdiagramm festgelegt.	Umfrage

Tabelle 5.31: Identifizierte Probleme im Zustandsdiagramm

Code	Problembeschreibung	Quelle
Zusammenhang mit Klassendiagramm	Es ist nicht klar, dass das Zustandsdiagramm auf die Attribute des Klassendiagramms zugreift.	
Zweck des Diagramms	Der Zweck des Diagramms wird nicht verstanden.	
Modellierung der Nutzersicht statt System- bzw. Objektsicht	Es wird die Nutzersicht statt der systeminternen Sicht modelliert.	

Code	Problembeschreibung	Quelle
Festlegen von Zuständen	Das Festlegen von Zuständen als Status eines Objekts fällt schwer bzw. das Festlegen von Zuständen im Gegensatz zu Zustandsübergängen (Actions) fällt schwer.	(Stikkolorum et al., 2018)
Verwechslung mit Aktivitätsdiagramm	Ein Zustand wird wie eine Aktivität behandelt und bezeichnet.	Umfrage
Ende des Lebenszyklus	Es ist nicht klar, wann der Lebenszyklus eines Objekts zu Ende ist.	
Logik	Die Darstellung der geforderten logischen Abfolge fällt schwer.	
Verbindungen zwischen Zuständen	Die Identifikation von Events, Guards und actions() einer Transition fällt schwer (Event [Guard] / action ()) bzw. die Unterscheidung zwischen Event und Guard fällt schwer.	(Stikkolorum et al., 2018)
Verzweigung	Es bestehen Probleme bei der Darstellung, wenn ein Zustand in zwei mögliche Folgezustände übergeht bzw. es mehr als einen Folgezustand geben kann.	
Aufbau des Diagramms im Allgemeinen	Studierende haben Probleme mit dem Aufbau des Diagramms im Allgemeinen.	Umfrage
Benennung Guard	Es ist nicht klar, wie ein Guard benannt werden soll.	
Detailgrad	Die Bestimmung des Detailgrades ist unklar.	Umfrage

Tabelle 5.32: Identifizierte Probleme im Sequenzdiagramm

Code	Problembeschreibung	Quelle
System unklar	Studierende haben den Verwendungszweck nicht verstanden und wissen nicht, was als System verstanden wird und wie dieses dargestellt wird.	Umfrage
Akteur	Akteure im Sequenzdiagramm sind unbekannt.	
Spezifikation bzw. Identifikation von Ausnahme-szenarien	Probleme beim Umgang mit Ausnahmeszenarien, z. B.: Was ist ein Ausnahmeszenario ein Fehlerfall?	
Festlegen der interagierenden Klassen (Objekte)	Die Vorgehensweise, wie man die Klassen identifiziert, die als Objekte miteinander interagieren sollen, ist unklar.	(Bolloju & Leung, 2006; Sien et al., 2010; Stikkolorum et al., 2018)
Aufbau des Diagramms (von links nach rechts von oben nach unten)	Die Leserichtung des Diagramms bzw. die Richtung des Nachrichtenversands ist unklar.	(Stikkolorum et al., 2018)
Logik und zeitliche Abfolge	Probleme beim Festlegen der Logik, welche Nachricht zuerst verschickt wird und welche nicht benötigt wird.	Umfrage
Verwendung von Fragmenten		Umfrage
> loop	Probleme beim Festlegen der Anzahl der Schleifendurchläufe, Probleme bei der Platzierung des loop-Fragments	
> opt	Probleme beim Festlegen der Bedingung für das opt-Fragment Probleme bei der Platzierung des opt-Fragments	

Code	Problembeschreibung	Quelle
> if/alt	Probleme beim Festlegen der Bedingungen für das alt-Fragment Probleme bei der Platzierung des alt-Fragments	
Interaktion zwischen Objekten mittels Methoden	Es ist unklar, dass die Interaktion zwischen Objekten mittels Methoden (Nachrichten) stattfindet.	
Instanzen (Objekte) keine Klassen	Es ist nicht klar, dass das Sequenzdiagramm ein Verhaltensdiagramm ist und deshalb objektbasiert modelliert wird und nicht klassenbasiert.	
Unterschied Akteur und Objekt	Die Unterscheidung zwischen Akteuren und Objekten fällt schwer.	
Verwendung der korrekten Pfeilrichtung	Die Verwendung der korrekten Pfeilrichtung fällt schwer.	
Verwendung der korrekten Pfeilart	Es ist nicht klar, welche Pfeilart verwendet werden soll.	
Geschachtelte Methodenaufrufe	Es fällt schwer geschachtelte Methodenaufrufe zu modellieren.	
Verwendung (a)synchroner Nachrichten	Es ist nicht klar, wie man entscheidet, welche Nachrichten synchron und welche asynchron gesendet werden.	(Stikkolorum et al., 2018)
Bedeutung synchrone Nachricht	Es ist nicht klar, wann eine Nachricht als synchron deklariert wird.	
(A)synchrone Nachricht (Notation)	Studierende haben Probleme damit, dass asynchrone Pfeile auf die Eine und synchrone Pfeile auf eine andere Art dargestellt werden.	

Code	Problembeschreibung	Quelle
Antwortnachricht	Es ist nicht klar, ob es immer eine Antwortnachricht geben muss.	(Stikkolorum et al., 2018)
Zusammenhang Klassen- und Sequenzdiagramm	Es ist nicht klar, dass Objekte für das Sequenzdiagramm aus dem Klassendiagramm stammen.	(Sien, 2011; Sien et al., 2010), Umfrage
Welche Klassen enthält die Methode	Es ist nicht klar, wie festgelegt wird, welche Methode (oder Funktion) zu welchem Objekt gehört.	
Fehlende Nachrichten	Es fehlen Nachrichten um das Szenario vollständig darzustellen.	(Bolloju & Leung, 2006)
Nachrichtensender und Methodenhalter	Es ist unklar, wie festgelegt wird, welches Objekt welche Methode hält (besitzt) und welches Objekt sie hingegen aufruft (also eine Nachricht schickt.	(Sien et al., 2010; Stikkolorum et al., 2018)
Identifikation der richtigen zu sendenden Nachricht	Es ist unklar, wie festgelegt wird, welche Nachricht verschickt werden muss.	(Sien et al., 2010; Stikkolorum et al., 2018)
Rolle des Aktors im Diagramm unklar	Akteure im Sequenzdiagramm sind unbekannt.	
Objekt oder Attribut	Es gibt Schwierigkeiten bei der Entscheidung, ob ein Element als interagierendes Objekt oder als aufgerufenes Attribut in einem Objekt modelliert werden soll.	
Reihenfolge der Entstehung der Objekte	Es ist unklar, welches Objekt erst durch einen create()-Aufruf entsteht und welches schon auf Basis der Aufgabenstellung da ist.	

Code	Problembeschreibung	Quelle
Ein oder mehrere Objekte von der gleichen Klasse	Es ist unklar, ob ein oder mehrere Objekte der gleichen Klasse angelegt werden können.	
Destroy- Objekt zerstören	Es ist unklar, wie die Zerstörung eines Objekts modelliert wird.	
Objektnotation	Studierende haben Probleme bei der Notation von Objekten und mit der Notation von statischen Klassen.	
Statische Klassen als Objekte	Die Repräsentation einer statischen Klasse ist unklar.	
Fehlende Objekte	Im Diagramm sind mehr Objekte notwendig als vorhanden.	(Sien, 2011)
Anlegen von Objekten	Das Anlegen von Objekten ist unklar.	
Angabe von Rückgabe-parametern	Rückgabetypen von Methoden werden angegeben, sollten aber in einem Sequenzdiagramm keine Rolle spielen. Dafür existieren (a)synchrone Nachrichtentypen.	
Umfang des Diagramms	Es ist nicht klar, wann das Diagramm vollständig ist.	(VanderMeer & Dutta, 2009), Umfrage
Vollständigkeit des Diagramms	Es bestehen Schwierigkeiten dabei, herauszufinden wann das Sequenzdiagramm laut der Aufgabenstellung abgeschlossen ist.	
Diagrammdetail-grad	Es ist unklar, wie viele Informationen in das Diagramm aufgenommen werden.	
Überblick behalten	Es fällt schwer, den Überblick über das Diagramm zu behalten (Welche Information ist enthalten, welche nicht.)	

5.3.5.1 *Qualitätssicherung*

Um die Aussagekraft dieser qualitativen Forschung zu sichern, werden im folgenden sechs Qualitätskriterien (unter Berücksichtigung von Kirk und Miller (1986), Flick (1987) und Kvale (1987)) nach Mayring (2016) betrachtet: Dokumentation der Methode, Interpretationssicherung, Nähe zum Objekt, Regelbegrenztheit, kommunikative Validierung und Triangulation:

DOKUMENTATION DER METHODE Qualitativ orientierte Forschung ist viel spezifischer auf das Thema bezogen als quantitative Forschung. Methoden werden speziell für dieses Thema entwickelt und differenziert. Deshalb muss der Forschungsprozess detailliert dokumentiert werden, um nachvollziehbar zu werden (Mayring, 2016). Dieses Kriterium wird mit dem vorliegenden Abschnitt, der Beschreibung der Erhebung und der Auswertung begründet.

GARANTIEN FÜR DIE AUSLEGUNG Bei diesem Kriterium ist zu beachten, dass Interpretationen nicht festgelegt werden müssen, sondern argumentativ begründet werden müssen (Hirsch, 1968; Mayring, 2016; Terhart, 1981). Es sollte zwischen verschiedenen Kriterien unterschieden werden:

- Das vorherige Verständnis muss angemessen sein; so wird die Interpretation von der Theorie geleitet (Hirsch, 1968; Mayring, 2016; Terhart, 1981). Die Kodierer sind nicht fachfremd in der Disziplin und haben daher ein adäquates Verständnis des Themas.
- Diese Interpretation muss in sich schlüssig sein; Brüche, wenn es sie gibt, müssen erklärt werden (Hirsch, 1968; Mayring, 2016; Terhart, 1981). Für alle zugeordneten Codes gibt es in einem Codebuch Notizen, die die Codes erklären und begründen. Viele der Daten wurden in vivo kodiert oder relativ nah an den Daten kodiert, so dass wenig Raum für Interpretationen bleibt.
- Es sollten alternative Interpretationen gesucht und geprüft werden. Die Widerlegung solcher „Negativfälle“ Becker und Geer (1979) oder negativer Interpretationen kann ein wichtiges Argument zur Rechtfertigung der Gültigkeit von Interpretationen sein (Mayring, 2016). Die Daten wurden paarweise kodiert. Eine paarweise Kodierung führt immer zu alternativen Interpretationen, die diskutiert werden müssen, bis ein Konsens erreicht und ein Code zugewiesen wurde.

DIE NÄHE ZUM OBJEKT In der qualitativen Forschung wird dies vor allem durch eine möglichst enge Verbindung zur Alltagswelt der untersuchten Personen erreicht. Ein qualitativer Forscher versucht, sich in das „Feld", die natürliche Welt der Probanden zu begeben, anstatt sie ins Labor zu bringen (Mayring, 2016). Die Studie wurde im Rahmen von vorlesungsbegleitenden (Labor-)Übungen durchgeführt. Sie wurden somit in den normalen Betrieb integriert. Mögliche Störfaktoren könnten die Aufnahmegeräte und Videokameras sein, die „unnatürlich" sind oder nicht dem normalen Betrieb entsprechen.

REGELBESCHRÄNKUNG Qualitative Forschung muss ihr Material systematisch bearbeiten und bestimmte Verfahrensregeln einhalten. Analyseschritte sollten im Vorfeld definiert werden, das Material in sinnvolle Einheiten aufgeteilt werden, damit die Einheiten in der Analyse systematisch bearbeitet werden können (Mayring, 2016). Es wurden verschiedene Evaluationsmethoden analysiert und eine Entscheidung für die strukturierende qualitative Inhaltsanalyse nach Mayring (z. B. Mayring (2016)) getroffen, der ein Modell für den Ablauf des Analyseprozesses vorgibt.

KOMMUNIKATIVE VALIDIERUNG Dieses Kriterium schlägt vor, die Validität der Ergebnisse zu überprüfen, indem sie den Forschern erneut präsentiert und mit ihnen diskutiert werden (Heinze & Thiemann, 1982; Klüver, 1997; Mayring, 2016). Wenn die Forscher die Analyseergebnisse und Interpretationen bestätigen, kann dies ein wichtiges Argument für die Validität der Ergebnisse sein (Scheele & Groeben, 1988). Dieses Kriterium wurde nicht umgesetzt.

TRIANGULATION Die Qualität der Forschung kann durch die Kombination mehrerer analytischer Ansätze verbessert werden (Denzin, 1989; Fielding & Fielding, 1986; Jack, 1979; Mayring, 2016). Denzin (1989) stellten dies auf verschiedenen Ebenen fest: Es können verschiedene Datenquellen herangezogen werden, unterschiedliche Interpreten, theoretische Ansätze oder Methoden. Zur Erfüllung der Triangulation ist folgendes zu nennen: Die Daten wurden paarweise kodiert, sodass vor der Zuweisung eines Codes ein Konsens erreicht werden musste. Das Paar bestand aus dem Versuchsleiter und einer Person, die mit dem Bereich nicht vertraut war. Der fachfremde Partner hinterfragte Zuweisungen, sodass eine Zuweisung reflektiert werden musste. Zwei weitere Codierer mit Fachgebietskenntnissen codierten zusätzlich Teile der Daten ohne weiteren Partner (mindestens ein Diagrammtyp). Dieser Prozess wurde durchgeführt, um Fehlinterpretationen und Fehleinschätzungen zu vermeiden und die Diskussion zu fördern.

Demnach sind fünf der sechs vorgeschlagenen Gütekriterien zur Sicherung der Qualität der Ergebnisse angewendet worden.

5.3.6 *Zusammenfassung und Ableitung der Maßnahmen zur Begegnung der Probleme*

Zusammenfassend lassen sich die in Abb. 5.5 und Abb. 5.6 dargestellte prozentuale Verteilung und absolute Häufigkeiten ableiten. Dabei ist zu beachten, dass die gesamte Anzahl der Findings, die in der Studie identifizierten Probleme darstellt, die explizit aufgeführten Findings zu „Quelle ausschließlich Studie“ zeigen diejenigen an, die nur in der Studie festgestellt wurden. Für das Sequenzdiagramm wurden insgesamt die meisten Problemstellen identifiziert, die geringste Anzahl im Zustandsdiagramm. Dies trifft auch für die Findings zu, die ausschließlich in der Studie festgestellt wurden. Aus Sicht der Literatur wurden bisher größtenteils Probleme im Use-Case-Diagramm und Klassendiagramm festgestellt.

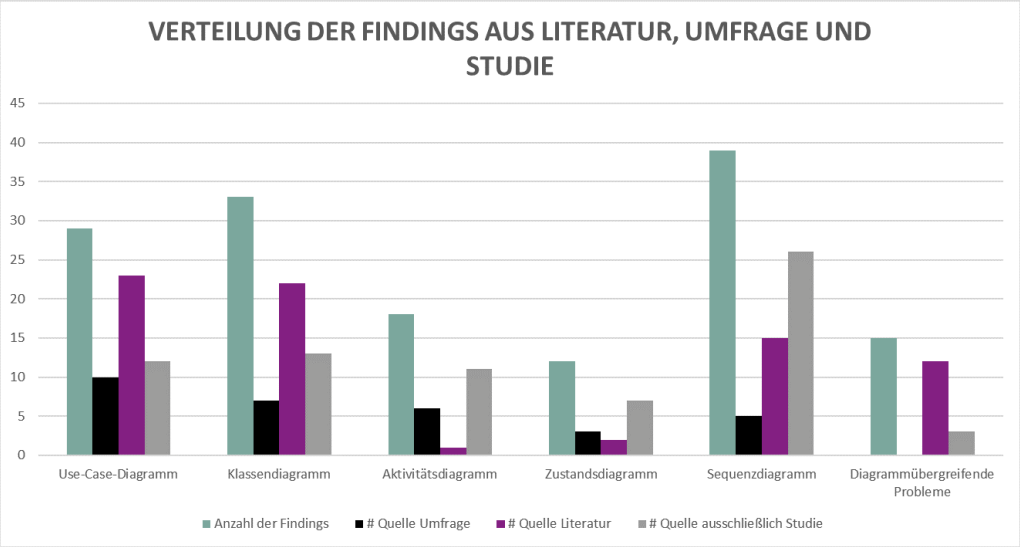


Abbildung 5.5: Absolute Häufigkeiten der identifizierten Probleme aufgeteilt in Findings in der Literatur, der Umfrage und der Studie

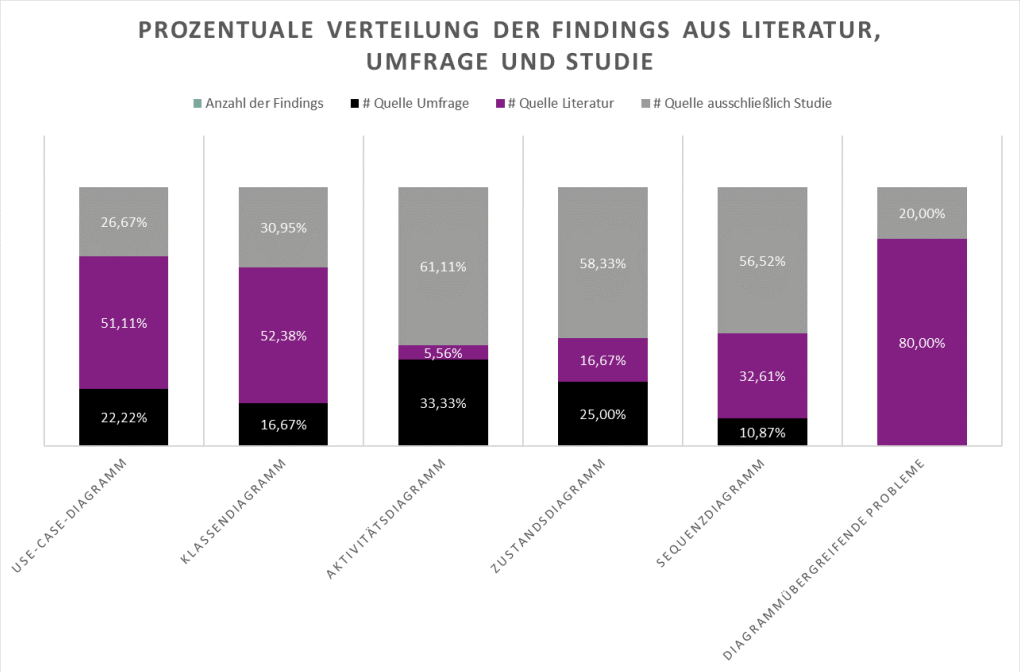


Abbildung 5.6: Prozentuale Verteilung der identifizierten Probleme aufgeteilt in Findings in der Literatur, der Umfrage und der Studie

Nach der Präsentation der Ergebnisse der Studie soll nun abschließend eine Bewertung der Schwierigkeiten bzw. Probleme Studierender im Modellierungsprozess, im Hinblick auf die Lernhindernisklassifikation und auf mögliche Anforderungen an eine Interventionsmaßnahme durch eine Softwarelösung, gegeben werden.

Bezogen auf die Lernhindernisklassifikation, vorgestellt in Abschnitt 3.2, lassen sich die diagrammspezifischen Probleme in die Klasse der epistemologischen Lernhindernisse einordnen. Probleme mit der Verwendung der Software sowie der Aufgabenbeschreibungen, werden der Klasse der didaktischen und der ressourcenbezogenen Lernhindernisse zugeordnet.

Es werden demnach Interventionsmaßnahmen benötigt, die einerseits inhaltliche Unterstützungsmöglichkeiten bieten sowie andererseits erleichterten Zugang und Bedienmöglichkeiten für die Modellierung. In Tabelle 5.33 werden Merkmale referenziert, die als Interventionsmaßnahmen abgeleitet wurden oder mit Hilfe derer Interventionen durchgeführt werden können.

Tabelle 5.33: Ableitung von Merkmalen für und als Interventionsmaßnahmen

Feature	Kurzbeschreibung
Vordefinierte Artefaktstruktur für SW-Designergebnisse	Ablage vom Projektartefakten in einer vordefinierten Struktur
Eye-Movement Modeling Example	Visualisierte Darstellung von Expertenstrategien, Darstellung von Live Beispielen
Tutorials (integrierter Browser)	Strukturierte Bereitstellung von externem Material, z. B. Videos, Screencasts etc.
Unterstützung bei der Identifikation von Use-Cases und Aktoren	Automatische Analyse von Textbausteinen und Präsentation von Vorschlägen
Unterstützung bei der Identifikation von Klassen und Methoden	Automatische Analyse von Textbausteinen und Präsentation von Vorschlägen
Einsatz von Erweiterter Realität	Unterstützung bei Abstraktionen, Ermöglichung von Perspektivwechsel
Allgemeiner Aufbau	Aufbau im Office-Stil für Wiedererkennungswert
Eingeschränkte Anzahl an Diagrammtypen	Angebot nur der in der Vorlesung behandelten Diagrammtypen
Erleichtertes Hinzufügen neuer Diagramme	Hinzufügen von Diagrammtypen analog zu Windows-Explorer
Strukturierte Bereitstellung von Unterlagen	strukturierte Bereitstellung von Vorlesungsunterlange, Beispielen und weiterführender Literatur in verschiedenen Formaten
Indizierbare Unterlagen	Angabe von Seitenzahlen für Unterlagen
Durchsuchbare Unterlagen	Unterlagen sind nach Begriffen durchsuchbar
5 Sterne Ranking	Unterlagen und Tutorials sind für alle Nutzer bewertbar, die Bewertung ist für alle Nutzer sichtbar

Feature	Kurzbeschreibung
Kommentarfunktion	Artefakte sind kommentierbar. Kommentare sind für alle Nutzer sichtbar
Speichern mit Kommentar/ Versionsverwaltung	Artefakte müssen bevor sie gespeichert werden kommentiert werden
Nutzerspezifische Hilfestellungen	Hilfestellungen können nutzerspezifisch empfohlen werden
Eingeschränkte Toolbox (Elemente) für Editoren	Editoren enthalten nur für das Diagramm notwendige Elemente und beispielweise keine besonderen Stereotypen
Eingeschränkter Funktionsumfang der Editoren	UML Syntax muss zum Teil händisch vervollständigt werden, Vervollständigung von Datentypen, Beschreibung der Transitionen
Explizit manuelle Verwendung der UML Syntax	UML Syntax muss zum Teil händisch vervollständigt werden, Vervollständigung von Datentypen, Beschreibung der Transitionen
Speichern in DB	Artefakte werden in einer Datenbank und nicht lokal abgelegt
Problemerkennung	Probleme/Regelverstöße in Diagrammen werden auf Nutzerwunsch erkannt
Problemspezifische Hilfestellungen	Unterlagen und Tutorials können problemspezifisch empfohlen werden
Nutzerspezifische Anwendung	Integration einer Nutzerverwaltung- Um individuelle Unterstützung gewährleisten zu können, ist die Anwendung mit einer Nutzerverwaltung ausgestattet.
Verwendung von Templates	Angebot strukturierter Dokumentation von Use-Cases mittels Templates
Versionierung	Versionierung zur Wiederherstellung von Artefakten

Nachfolgend werden nun Möglichkeiten zur Realisierung dieser Interventionsmaßnahmen im Verlauf der Arbeit experimentell, prototypisch vorgestellt.

6

DESIGN: KONZEPTION EIGENER SCAFFOLDS

One of the most important tasks of the teacher is to help his students. This task is not quite easy; it demands time, practice, devotion, and sound principles.
The student should acquire as much experience of independent work as possible. But if he is left alone with his problem without any help or with insufficient help, he may make no progress at all.
If the teacher helps too much, nothing is left to the student. The teacher should help, but not too much and not too little, so that the student shall have a reasonable share of the work.

— George Pòlya (Polya, 1985, S. 1)

Nach der Identifikation der Schwierigkeiten Studierender während der Modellierung mit der UML, sollen nun im folgenden Kapitel inhaltliche Hilfsmittel entwickelt werden, die die Bedürfnisse der Studierenden respektieren und dabei, zu deren Unterstützung, den Einsatz aktueller Technologien bewerten und, wo möglich, bereitstellen.

Verortet im Prozess des User Centered Design (UCD, siehe Abb. 6.1), definiert in der ISO 13407, die 2010 durch die ISO 9241-210 ersetzt wurde (DIN Deutsches Institut für Normung e. V. (2020)), ist die Identifikation der Schwierigkeiten der Phase des Analysierens zuzuordnen, während dieses und das folgende Kapitel die Phase des Entwerfens und Produzierens vorstellen. Die Identifikation der Schwierigkeiten diente dazu, den Kontext zu verstehen. Dazu wurde in Abschnitt 5.3.5 der Problemkatalog angefertigt. Auf Basis des Katalogs wurden zudem erste Maßnahmen als Fazit (siehe Abschnitt 5.3.6) abgeleitet. Diese werden nun hier weitergeführt. In der Phase des Entwerfens des UCD geht es darum, zu konzipieren, zu gestalten und Prototypen zu entwerfen. Es werden Gestaltungslösungen entwickelt, die die oben abgeleiteten Anforderungen erfüllen. Dieses Kapitel endet mit einem Überblick über die Literatur zu Unterstützungsmöglichkeiten (Scaffolds) im Bereich der

Software Engineering-Lehre (siehe Abschnitt 6.13) und der pädagogisch-didaktischen Verortung des Konzepts der Scaffolds in Form eines Rückbezugs (siehe Abschnitt 6.12).

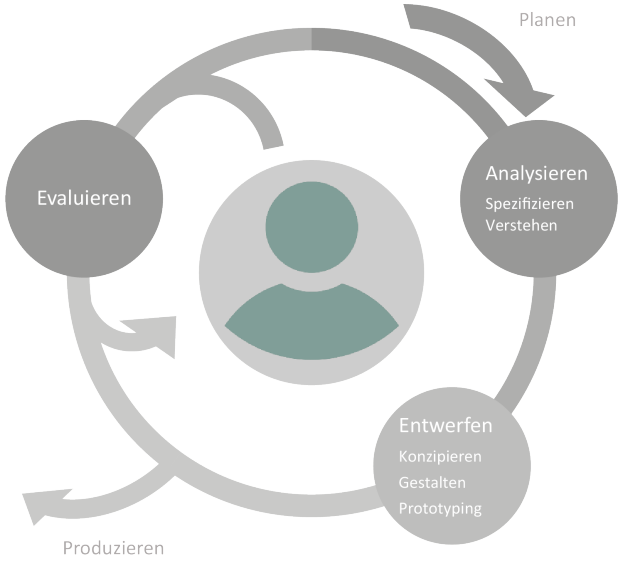


Abbildung 6.1: User Centered Design Process adaptiert aus Fuchs (2020) und DIN Deutsches Institut für Normung e. V. (2020)

In diesem Rahmen wird die Forschungsfrage FF2 mit den Unterfragen FF2.1-FF2.3 beantwortet:

- FF2: Welche Unterstützungsmöglichkeiten existieren bei der Problemlösung?
  - FF2.1: Welche Arten von Unterstützung für Software Engineering und speziell für die Modellierung gibt es?
  - FF2.2: Wie können diese Unterstützungsmöglichkeiten experimentell technologiebasiert angeboten werden?
    * FF2.2.1: Ist eine Technologie wie Augmented Reality als Unterstützungsmöglichkeit geeignet?
    * FF2.2.2: Inwiefern ist ein Eye-Movement Modeling Example als Unterstützungsmaßnahme geeignet?
  - FF2.3: Wie können diese Unterstützungsmöglichkeiten angeboten werden?
    * FF2.3.1: Wie können Unterstützungsmöglichkeiten individuell angeboten werden?
    * FF2.3.2: Inwiefern lassen sich Hilfsmittel zwischen Dozierendem und Studierendem und unter Studierenden empfehlen?

Die Unterstützungsmaßnahmen wurden auf Basis der in Abschnitt 5.3.2 beschriebenen Fragebogenstudie und auf Basis des in Abschnitt 5.3.5 aufgestellten Katalogs über die bestehenden Schwierigkeiten Studierender sowie durch gewonnene Erkenntnisse durch eine Literaturstudie konzeptioniert. Scaffolds, auf der Basis von Augmented Reality, Eyetracking und Textanalysetechniken unter Verwendung von Natural Language Processing (NLP), stellen zudem aktuelle technische Ansätze dar, die in den folgenden Abschnitten beschrieben werden. Zur Evaluation dieser Scaffolds werden Ziele, Strategien und Maßnahmen erfasst. Dabei sei angemerkt, dass Ziel dieser Arbeit die Einbindung verschiedener Scaffolds innerhalb einer Modellierungsumgebung (siehe Abschnitt 6.1) ist. Konkrete Inhalte werden für die Scaffolds von Dozierenden aber selbst bestimmt. Im Rahmen der experimentell evaluierten Scaffolds wurden spezifisch Lernsettings konzipiert, sodass hier noch keine Rückschlüsse auf andere Inhalte gezogen werden sollten.

Abbildung 6.2 zeigt den relevanten Ausschnitt aus Abb. 3.5 und gibt einen Überblick über die im Rahmen der Arbeit konzipierten Scaffolds: Eingebettet in die Theorie der konstruktivistischen Lernumgebung lassen sich drei Kernelemente (eine Umgebung (Abschnitt 6.2), klassische Scaffolds (Abschnitte 6.3 bis 6.8) und technologiebasierte Scaffolds (Abschnitte 6.9 bis 6.11)) festmachen. Zentrales Element ist eine alternative Modellierungsumgebung (Ariadne metaphorisch gezeigt als das zentrale Fortbewegungsmittel), die integrierte Scaffolds bereitstellt (metaphorisch gezeigt als Navigationselemente). Zu den konzipierten Scaffolds gehören klassische Hilfsmittel und solche, die auf Basis neuer Technologien realisiert wurden. Allen Scaffolds gemein ist eine gemeinsame Datenbasis, die genutzt wird, um entsprechende Inhalte bereitzustellen. Im Sinne einer studierendenzentrierten Lehre ist dies unverzichtbar, da Studierende mit Hilfe der gemeinsamen Datenbasis ohne zusätzlichen Aufwand beispielweise Technologien (z. B. PC und AR-Brille) wechseln können.

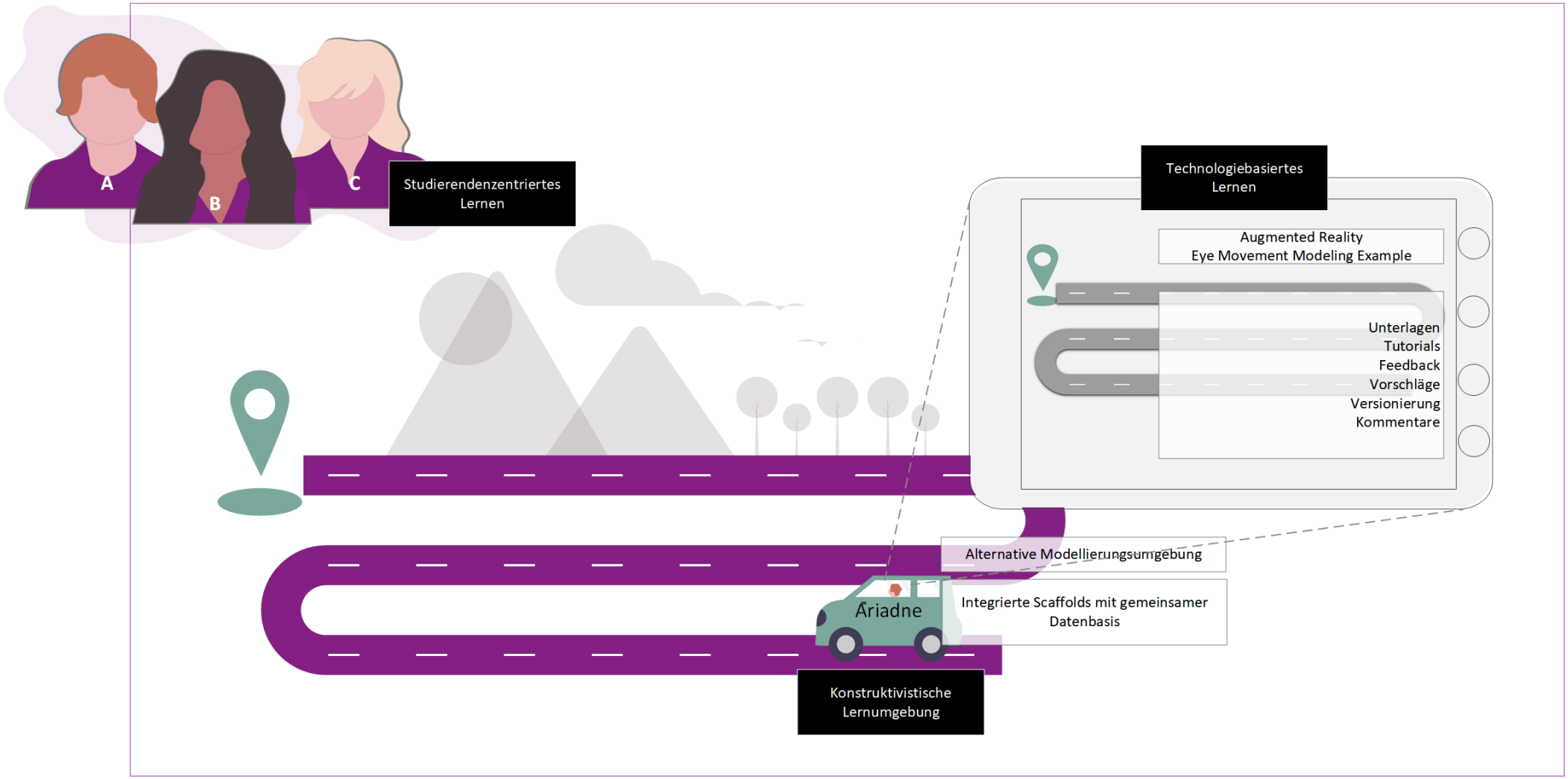


Abbildung 6.2: Überblick über die konzipierten Scaffolds

6.1 Ziele, Strategien und Massnahmen zur Konzeption von Scaffolds

Im Wesentlichen haben die konzipierten Scaffolds drei Wurzeln:

- Ursprung in der Fragebogenstudie
- Ursprung in den Daten, mit dem Fokus auf der Adressierung spezifischer beobachteter Probleme, zum Teil experimentell didaktisch aufbereitet
- Ursprung in der Literatur

Natürlich stellt sich die Frage, warum diese Hilfsmittel in einer Umgebung zur Verfügung gestellt werden müssen, wenn es doch Zugang zum Internet gibt. Dazu gibt es mehrere Gründe:
Ein vermuteter Grund liegt darin, dass Studierende häufig nicht hinterfragen, welche Quellen sie für ihre Modellierung heranziehen können oder sollen, weshalb ein „Empfehlungssystem“ integriert wurde, welches es sowohl Studierenden wie auch Dozierenden erlaubt, Unterlagen und Tutorials zu bewerten und vor allem schon vordefinierte, wichtige Textstellen auszuwählen. Ein weiterer Grund liegt darin, dass Studierende nicht den Aufwand betreiben, Vorlesungsunterlagen zu durchsuchen. Auch hierfür ist die Empfehlung des Hilfsmittels mit Seitenangabe hilfreich und soll dazu führen, dass ressourcenbezogene Probleme minimiert werden. Laut der Cognitive Load Theory (CLT) (siehe Abschnitt 3.2.1) sollte zudem Lernmaterial so konzipiert werden, dass die inhaltliche (intrinsic) und äußere (extraneous) kognitive Belastung beim Lernen reduziert bzw. kontrolliert wird (Schmidt-Weigand, Franke-Braun & Hänze, 2008). Die Reduktion der äußeren Belastung wird unter Anderem mit Hilfe der Integration der Scaffolds direkt in die Modellierungsumgebung intendiert.

Studierende erkundigen sich häufig nach der Korrektheit eines von ihnen erstellten Diagramms. Gerade in der Modellierung ist dies nicht trivial zu bewerten und hängt insbesondere vom jeweiligen Designer und von subjektiven Gegebenheiten ab, d. h. das Problem für die Studierenden wird durch die Tatsache noch verschärft, dass sie, wenn sie Hilfe suchen, keine eindeutigen Ratschläge, manchmal sogar widersprüchliche Aussagen erhalten und dies möglicherweise nicht nur von verschiedenen Personen, sondern von der gleichen Person (Thomasson et al., 2006a). Deshalb wurde prototypisch eine „Problemerkennung“ integriert, um Studierenden die Möglichkeit zu bieten, einerseits selbstständig und andererseits systematisch immer gleiche Probleme zu identifizieren. Probleme sind dabei nicht gleichbedeutend mit Fehlern im Sinne des UML Metamodells, sondern vielmehr mit einem Verstoß gegen Regeln und

Richtlinien, die im Rahmen einer Studie erfasst wurden. Der Problemkatalog bildet die Basis für die Integration spezifischer Hilfsmittel für die Modellierung eines Softwaresystems.

Gerade bei Use-Case-Diagrammen und Klassendiagrammen stießen die Studierenden häufig auf die Fragestellungen wie Use-Cases oder Klassen identifiziert würden. Speziell für diese beiden Diagrammtypen und für dieses Problem wurden deshalb zwei Hilfsmittel integriert. Diese bieten auf Basis von bereits existierenden Projektbeschreibungen die Hilfestellung, relevante Textabschnitte aus den Beschreibungen für mögliche Use-Cases und in einem weiteren Schritt Vorschläge für Use-Cases zur Verfügung zu stellen. Weiterhin gibt es das Angebot, relevante Textabschnitte auf Substantive und Verben hin zu analysieren, um einem Nutzer die Identifikation von Klassenkandidaten und möglichen Methoden zu erleichtern. Operationalisiert wurden diese Analysen mit Hilfe von NLP.

Zu den Hilfsmitteln, die experimentell evaluiert wurden, zählen die Entwicklung eines UML-Klassendiagramm Editors mit Augmented Reality, sowie die Entwicklung eines Visualisierungstools mit Augmented Reality für ein spezifisches Lernszenario, um abstraktes Denken zu unterstützen und die Entwicklung sogenannter Eye-Movement Modeling Example (EMME)s, zur Erstellung von Sequenzdiagrammen. Dabei sehen Studierende ein Video, das neben auditiver Unterstützung auch die visuelle Unterstützung durch Blickbewegungen eines Experten enthält. Dass Erklärungen von Experten als erfolgreicher Scaffold dienen können, wurde auch von Feldgen und Clua (2012), Shabo, Guzdial und Stasko (1996), Linn und Clancy (1992), Linn (1992), evaluiert. Dies kommt auch dem Wunsch der Studierenden (aus dem Fragebogen) nach einem Livebeispiel entgegen (Jarodzka et al., 2012; Mason, Pluchino & Tornatora, 2015; Sweller, 1988).

Mit den Hilfsmitteln, die Augmented Reality nutzen, sollten neben ressourcenbezogenen auch didaktische Probleme adressiert werden. Bezogen auf den erstellten Katalog: Das Problem einen geeigneten Grad an Abstraktion für ein dediziertes Modell zu finden, das sich über die Diagramme hinweg hat identifizieren lassen, soll darüber adressiert werden, dass der Prototyp des zu entwickelnden Produkts bereits visuell vorhanden ist. Das Problem des „Überblick Behaltens“ sollte damit unterstützt werden, dass eine sehr große Modellierungsfläche im realen Raum zur Verfügung steht.

Durch Neukonzeptionen sollten im Wesentlichen neue Zugänge durch Expertenblicke oder erweiterte Perspektiven zum Lerngegenstand geschaffen werden.

6.1.1 *Ergebnisse aus der Fragebogenstudie: Wünsche der Studierenden zu Scaffolds*

Wie einleitend angedeutet und zu einem Teil bereits in Abschnitt 5.3.2 beschrieben, wurde zusätzlich zur Think-Aloud Studie eine Fragebogenstudie für jeden bearbeiteten Diagrammtyp durchgeführt, in der die Studierenden neben den bereits beschriebenen Items zu bestehenden Schwierigkeiten auch nach Hilfsmitteln gefragt wurden, die sie sich zum Erlernen des Modellierens gewünscht hätten bzw. wünschen würden.

Konkret wurde folgendes Item eingesetzt, um die gewünschten Hilfsmittel der Studierenden zu erfragen:

> **Was hätte Sie beim Lösen der Aufgabe/bei der Modellierung Ihres [Diagrammart] unterstützt? (Auch unabhängig von eventuell aufgetretenen Problemen)**

Die Antworten wurden thematisch geclustert und es ergab sich folgende Aufstellung:

- „Andere“ IDE
- Einführung in IDE
- Dokumente
  - Vorlesungsunterlagen
  - Definitionen
  - Spickzettel
- Beispiele
  - Einfach
  - Komplex
  - Besondere Fälle
  - Live-Beispiel („Gedanken des Professors, wenn er eine solche Aufgabe bearbeitet“)
- Tutorials
- Internet
- Explizite Aufgabenbeschreibungen
- Feedback
  - „schnelles kurzes Feedback“
  - Antworten auf die Frage „Ist das in Ordnung?“
- Erläuterungen des Professors in konkreter Situation
- Musterlösungen
- Vorschläge

- Zusammenarbeit und Hilfe von Gruppenmitgliedern

Zusammenfassend lässt sich hier schließen, Studierende wünschen sich verschiedene Typen von Beispielen, wie einfache und komplexe Beispiele, aber auch Spezialfälle und Live-Beispiele. Außerdem wurden Tutorials, Lösungsbeispiele, Dokumente, zusätzliche Definitionen, Spickzettel und Feedback gewünscht. Insbesondere wünschen sie sich eine andere Modellierungssoftware sowie eine Einführung in die Modellierungssoftware bzw. daraus abgeleitet eine Modellierungssoftware, deren Bedienung leicht erlernbar oder intuitiv ist.
Die Auswertung trägt dazu bei Forschungsfrage 2.1 unter Einbezug eigener empirischer Daten zu beantworten.

6.1.2 *Hilfsmittel auf Basis aktueller Technologien*

Die Erhebungen der Bedürfnisse der Studierenden zu den Hilfsmitteln, die sie sich während der Modellierung zur Unterstützung wünschen, ist als Grundlage für die Konzeption solcher zu werten. Daher sollte diesen Bedürfnissen besondere Aufmerksamkeit gewidmet werden und Hilfsmittel auf dieser Basis konzipiert werden. Trotz Konjunktiv in der Frage „Was hätte Ihnen geholfen?“ werden keine visionären Hilfsmittel genannt, sondern „Altbekanntes“, d. h. im Gedächtnis manifestierte Hilfsmittel, wie Beispiele oder Unterlagen. Allerdings ist es gerade in einer Disziplin wie Software Engineering notwendig, sich mit ständig weiterentwickelnden Technologien auseinanderzusetzen. Deshalb wurden zusätzlich zu den Hilfsmitteln, die auf Basis der Bedürfnisse der Studierenden konzipiert wurden, auch technologiebasierte statische Scaffolds (mit Hilfe von Eyetracking, Augmented Reality und NLP) konzeptioniert, die im Bereich der Lehr-Lernforschung zeitgemäß sind und dem aktuellen Forschungsstand entsprechen. Der abgeleitete Problemkatalog (Abschnitt 5.3.5) bildete dabei eine Grundlage.

6.1.2.1 *Einsatz Erweiterter Realität (Augmented Reality)*

Eine Möglichkeit wurde in der Verwendung erweiterter Realität, sogenannte Augmented Reality (AR), vermutet. Akçayir et al. (2016) und Bacca, Baldiris, Fabregat, Graf und Kinshuk (2014) liefern bereits systematische Literaturreviews, die vielversprechende Ansätze zum Einsatz von AR in der Lehre listen. Eine steigende Anzahl an Publikationen zu AR im Lehreinsatz ist seit 2013 zu verzeichnen (Akçayir et al., 2016, S.4). Zu Software Engineering, speziell Software Design, gibt es keine gelisteten Veröffentlichungen. Beide Literaturreviews untersuchen aber

Vorteile und Nachteile von AR, die zur Entscheidung den Einsatz von AR als Hilfsmittel ebenfalls zu evaluieren, beigetragen haben.

Tabelle 6.1: Vorteile von AR in der Lehre laut Akçayir et al. (2016, S.141 (übersetzt)). (Betrachtete Vorteile sind **fett** markiert.)

Induktive Kategorien	**Subkategorien**	**Anzahl**	**Beispielforschung**
Lerner-Outcomes	**Verbessertes Lernergebnis**	32	Lu Liu, 2015
	Verbesserte Lernmotivation	10	Chiang et al., 2014a
	Hilft Studierenden beim Verstehen	7	Kamarainen et al., 2013
	Sorgt für positive Einstellung	6	Wojciechowski Cellary, 2013
	Steigerung der Zufriedenheit	4	Han et al., 2015
	Mindert Cognitive Load	2	Santos et al., 2014
	Stärkt Vertrauen	2	Lu Liu, 2015
	Verbessert räumliches Denken	2	Lin, Chen, Chang, 2015
Pädagogische Beiträge	Steigert die Freude	8	Ibanez et al., 2014
	Steigerung des Engagements	6	Liu Tsai, 2013
	Steigert Interesse	4	Zhang et al., 2014
	Bietet Kollaborationsmöglichkeiten für Studierende	3	Lin, Duh, Li, Wang, Tsai, 2013
	Ermöglicht Kommunikation zwischen Studierenden und Dozierenden	2	Zarraonandia et al., 2013
	Fördert Selbststudium	2	Ferrer-Torregrosa et al., 2015
	Kombiniert physikalische und virtuelle Welten	1	Dunleavy et al., 2009

	Gestattet Lernen nach Learning by doing	1	Hsiao et al., 2012
	Studierendenzentrierte Technologie	1	Kamarainen et al., 2013
	Ermöglicht multisensorisches Lernen	1	Lu Liu, 2015
	Ermöglicht Lernen schnell Informationen zu erhalten	1	Chiang et al., 2014b
Interaktion	Bietet Interaktionsmöglichkeiten (Studierender- Studierender)	4	Kamarainen et al., 2013
	Studentisches Material	2	Lin et al., 2011
	Studierender-Dozierender	1	Zarraonandia et al., 2013
Andere	**Ermöglicht die Visualisierung unsichtbarer Konzepte, Ereignisse und abstrakter Konzepte**	5	El Sayed et al., 2011
	AR ist für Studenten zu verwenden	4	Di Serio et al., 2013
	Reduziert Labormaterialkosten	1	Ferrer-Torregrosa et al., 2015

Bezogen auf die bestehenden Lernhindernisse und Probleme Studierender sollten die folgenden von Akçayir et al. (2016) (siehe Tabelle 6.1) identifizierten Vorteile von AR für die Arbeit ausgenutzt werden:

- Durch einen völlig neuen Zugang zum Lernmaterial sollen didaktische Lernhindernisse[25] adressiert werden. Bezogen auf die Cognitive Load Theory (Abschnitt 3.2.1) gilt es den Extraneous Load und den Intrinsic Load niedrig zu halten um Schwierigkeiten mit dem Lernmaterial zu kontrollieren und im Besten Fall auch zu reduzieren.
- AR kann laut Akçayir et al. (2016) (siehe Tabelle 6.1) dazu dienen, bessere Lernergebnisse zu erzielen und die Motivation beim Lernen zu erhöhen.

25 Diese Lernhindernisdimension (siehe auch Abschnitt 3.2) wurde aus der Cognitive Load Theory abgeleitet. Sie bezieht sich auf den Extraneous Load. Sie beschreibt die externen Einflüsse bezüglich der Struktur, dem Lernsetting.

- AR kann Studierenden helfen Lerninhalte zu verstehen (Akçayir et al., 2016); siehe Tabelle 6.1, indem es beispielsweise unsichtbare Konzepte, Ereignisse, und abstrakte Konzepte visualisierbar macht.

Nach der Einführung der übrigen experimentellen Scaffolds, werden in den nächsten Abschnitten zwei Applikationen vorgestellt und evaluiert, die die gewählten Vorteile von AR ausnutzen aber spezifisch für die Lehre von Softwaredesign mit der UML ausgerichtet sind. Eine Applikation hat zum Ziel einen anderen Zugang zu einer Modellierungsumgebung zu schaffen und damit Motivation und Lernergebnis in der Modellierung von Klassendiagrammen zu steigern.

Die andere Applikation adressiert den letztgenannten Punkt und versucht Anforderungen an eine Software und deren Design durch Visualisierungen abstrakter Konzepte und Ereignisse, die für die Studierenden relevant sind, sichtbar zu machen.

6.1.2.2 *Einsatz von Eye-Movement Modeling Examples*

Der Grundgedanke in der Konstruktion dieses Hilfsmittels besteht darin, Studierende durch die Bereitstellung von Eye-Movement Modeling Examples (EMMEs) zu unterstützen. EMMEs sind eine besondere Art der videobasierten Modellierung (Jarodzka et al., 2012, S.815):

> „EMMEs are worked-out examples of how an expert model performs a perceptual task, where a video case [..] is augmented with two types of procedural information. This information is gained from asking an expert to explain verbally how he or she performs the task based on watching the video case as well as from recording his or her eye movements whilst performing the task“

Ein solches EMME-Video kann Lernenden als Unterstützung angeboten werden, beispielsweise als Form des Lernens an Beispielen (Mason et al., 2015; Sweller, 1988). Ein EMME-Video besteht aus der vom Modell ausgeführten Aufgabe und optional der verbalen Erklärung des Modells, die die kognitiven Prozesse des Modells darstellt. Zusätzlich werden die Augenbewegungen des Modells dem Video überlagert, das die Wahrnehmungsprozesse des Modells darstellt (Jarodzka et al., 2012). Konkret angewandt auf den vorliegenden Anwendungskontext, einem Softwaredesign mit der UML, konstruiert das EMME-Modell beispielsweise ein Use-Case-Diagramm aus textuellen Anforderungen (textuelle Beschreibung der Anforderungen an die Software). Das Modell gibt während

der Konstruktion des Diagramms verbale Erklärungen. Die Kombination aus verbalen Erklärungen und visueller Anleitung soll die Studierenden dabei unterstützen, sich ihrer Schwierigkeiten bewusst zu werden, z. B. bei der Definition relevanter Anwendungsfälle.

Zusätzlich zu dem instruktionsunterstützenden Ziel dieser Studie über EMMEs im Bereich der UML, leistet die Evaluation dieses Hilfsmittels einen Beitrag zum Verständnis über die Wirksamkeit von EMMEs. Mit Hilfe einer Eyetracking-Studie wird beispielsweise nachgewiesen, dass es gelingt die Blicke der Studierenden auf den Blick des Experten zu lenken. Außer einer Studie aus dem Bereich Softwaretechnik (Bednarik, Budde & Vrzakova, 2018), die sich auf Codeverständnis und Lesestrategie konzentriert, sind keine weiteren Forschungen zu EMMEs vor allem im Bereich der Software-Engineering Ausbildung bekannt.

Im Folgenden werden die konkreten Umsetzungen, die auf Basis der bisherigen Erkenntnisse und der abgeleiteten Maßnahmen implementiert wurden, vorgestellt.

6.1.2.3 *Einsatz von Textanalysetechniken auf Basis von NLP*

Die Erstellung von Use-Cases ist der erste Schritt der Modellbildung nach dem Makroprozess und damit eine Form der Abstraktion (Balzert & Balzert, 2009, S. 27). Eine Identifikation von Use-Cases geschieht in der Regel auf Basis von Anforderungsdokumenten, beispielsweise auf Basis des Vision-Dokuments, das z. B. im Rational Unified Process vorgeschlagen wird (Kruchten, 2004; Rational Software Corporation, n. d.). Dies und auch die Identifikation von Klassenkandidaten und Methoden auf Basis von Beschreibungen von Use-Cases, erfordert eine Analyse natürlichsprachlicher Dokumente, bei der Studierende unterstützt werden sollen. Um diese Unterstützungsmaßnahme computergestützt zu operationalisieren wird eine Form von Natural Language Processing (NLP) eingesetzt. Ziel von NLP ist es, Maschinen dazu zu bringen, unsere gesprochene und geschriebene Sprache zu verstehen (Ganegedara, 2018, S. 2; Arumugam & Shanmugamani, 2018). Eine Vielzahl von Anwendungen heutzutage nutzt NLP, so z. B. Google Translate, Cortana oder Apple Siri. Aufgaben von NLP werden in Abschnitt 7.1.4.6 beschrieben. Um Klassenkandidaten zu identifizieren, können die Funktionen des NLP, Tokenisierung (Jurafsky & Martin, 2019, S. 15) und das sogenannte Part-of-Speech(Pos)-Tagging (Jurafsky & Martin, 2019, S. 133) angewandt werden. OpenNLP[26], eine frei verfügbare Implementierung zur Sprachverarbeitung, stellt bereits trainierte Modelle zur Verfügung um deutschsprachigen Text computergestützt zu verarbeiten und Substantive und

26 https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html

Verben, die für die Abbott-Technik detektiert werden müssen, maschinell zu identifizieren. Es wird dabei keine vollständige Satzanalyse benötigt, sondern es werden lediglich Wortkategorien bestimmt — eine Technik, die gut etablierte und solide Ergebnisse liefert. Um die Identifikation von Use-Cases zu unterstützen, werden die Funktionen des NLP der Satzsegmentierung (Jurafsky & Martin, 2019, S. 22) und der Tokenisierung (Jurafsky & Martin, 2019, S. 15) verwendet (Ganegedara, 2018; Jurafsky & Martin, 2019, S. 2; Arumugam & Shanmugamani, 2018). Dabei wird ein Textcorpus[27] in Bestandteile zerlegt, z. B. in einzelne Sätze oder Worte. Sätze, die beispielsweise einen Imperativ enthalten, können so detektiert werden. Für die Implementierung dieses Scaffolds wird angemerkt, dass sich diese Arbeit nicht mit der Optimierung oder Verbesserung von Forschungsansätzen zu NLP beschäftigt, aber bestehende Möglichkeiten ausnutzt, Studierenden erste Ansätze zur Analyse von Use-Cases und Klassenkandidaten zu bieten.

6.2 Ariadne als neue Modellierungsumgebung für Einsteiger

Die Beobachter der beschriebenen Think-Aloud Studie stellten Schwierigkeiten im Umgang mit der Modellierungsumgebung fest, weshalb zunächst damit begonnen wurde, eine neue Umgebung (Ariadne) zu realisieren, in der Novizen intuitiv modellieren können und nicht durch Probleme in der Verwendung der Umgebung daran gehindert werden, zu modellieren, sondern vielmehr motiviert werden, zielstrebig eigene Modelle zu entwerfen. Dem Ansatz von Sutch (2007) entsprechend, ist das Design insgesamt in Analogie zu bereits bekannten Office Produkten angelegt. Sie beschreiben, dass durch die Verwendung von Analogien und Metaphern den Teilnehmern geholfen werden kann, Muster bei der Problemlösung zu erkennen und so die Spannung beim Erlernen von etwas Neuem zu verringern und Vertrautheit in eine neue Situation zu bringen. Neue Aufgaben mit Vertrautem zu verknüpfen, trägt dazu bei, den Angstfaktor, den viele Menschen gegenüber einer Software empfinden, abzubauen und sie angenehmer zu gestalten (Sutch, 2007, S.334).

Bezogen auf die von den identifizierten Schwierigkeiten abgeleiteten Maßnahmen, können folgende High-Level-Anforderungen an eine Modellierungsumgebung definiert werden:

27 Jurafsky und Martin (2019) definiert Corpus (Plural Corpora) als maschinenlesbare Sammlung von Text oder Sprache.

- Die Software muss eine vordefinierte Struktur zur Ablage der Designergebnisse haben.
- Die Software muss bei Studierenden ein Gefühl der Wiedererkennung erzeugen.
- Die Software sollte einen eingeschränkten Funktionsumfang bereitstellen.
- Die Software muss Editoren für die in der Vorlesung behandelten Diagrammtypen zur Verfügung stellen.
- Die Software sollte die Möglichkeit bieten einem Projekt neue Diagrammtypen intuitiv hinzufügen zu können.

Für die prototypische Implementierung der IDE wurden konkret folgende Designentscheidungen getroffen:

- Die Software wird angelehnt an den Microsoft Office Stil designed und erhält ein ähnliches Bedien-, Farb- und Funktionskonzept.
- Die Navigationsstruktur für ein Projekt ist eingeschränkt und prinzipiell nach dem Vorschlag aus Lehrbüchern, wie in Balzert und Balzert (2009) und Rupp und die SOPHISTen (2014) ausgearbeitet, sodass Zusammenhänge zwischen einzelnen Diagrammen und der vorgelagerten Anforderungsanalyse deutlich werden.
- Es werden insgesamt fünf verschiedene Editoren für Diagrammtypen implementiert: Use-Case-Diagramm, Klassendiagramm, Aktivitätsdiagramm, Zustandsautomat und Sequenzdiagramm. Diese wurden ausgewählt, da sie zu den am häufigsten verwendeten Diagrammtypen im industriellen Einsatz und damit auch zu den am häufigsten im Lehrbetrieb eingesetzten, zählen (z. B. Anke & Bente, 2019; Dobing & Parsons, 2008; Erickson & Siau, 2007; Langer, Mayerhofer, Wimmer & Kappel, 2014; Reggio, Leotta, Ricca & Clerissi, 2013).
- Die einzelnen Editoren bieten jeweils Basiselemente. Das schließt Spezialfälle, wie beispielsweise individuelle Stereotype aus, inkludiert jedoch alle für die grundlegende Modellierung notwendigen Elemente.
- Gleichzeitig wurde bewusst davon Abstand genommen, UML Syntax vorzugeben bzw. automatisch zu erzeugen. So müssen beispielsweise Zustandsübergänge (Transitionen) mit Ereignis, Bedingung und Aktion manuell in ihrer korrekten Syntax hinzugefügt werden. Ein Automatismus, wie beispielsweise die Erzeugung des „/“ zwischen Bedingung und Aktion erfolgt nicht. Die Software bietet also Grundfunktionalität und erfordert in jedem Fall eine Komplettierung durch den Nutzer.

Die Navigationsstruktur ist so angelegt, dass zunächst eine Projektbeschreibung bzw. Produkt-Vision (in natürlicher Sprache) für jedes neu angelegte Projekt vorgesehen ist. Weiterhin wird bei Anlage des Projekts ein Use-Case-Diagramm (Abb. 6.6) angelegt, in dem Use-Cases und Akteure für das Projekt modelliert werden können. Es können ebenfalls Use-Cases in der Navigationsstruktur angelegt werden. Für alle Use-Cases werden eine natürlichsprachliche Beschreibung (Abb. 6.4), sowie ein Use-Case Template (siehe Abschnitt 4.4, Abb. 6.5) zur strukturierten Beschreibung und ein Klassendiagramm Editor (Abb. 6.7) angelegt. Unterhalb des Klassendiagramm-Editors ist es nun möglich weitere Diagramm-Editoren und beliebig viele Diagramme zu modellieren. Verfolgt wird damit der Makroprozess zur Erstellung eines OOA-Modells, wie in Abschnitt 4.3 beschrieben.

Abbildungen 6.4 bis 6.10 zeigen die einzelnen Editoren, wie sie prototypisch realisiert werden.

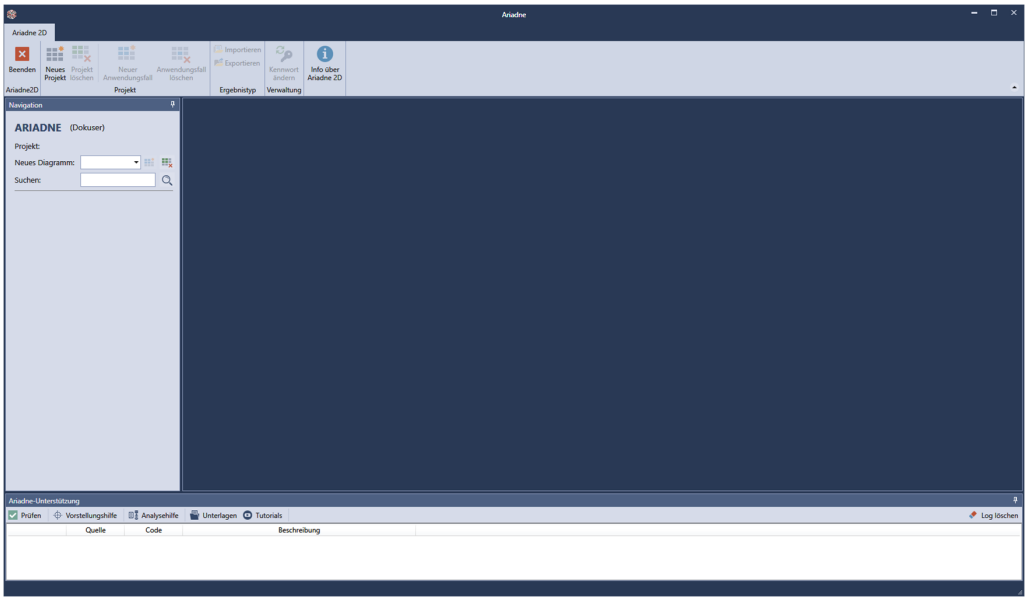


Abbildung 6.3: Oberfläche der Modellierungsumgebung

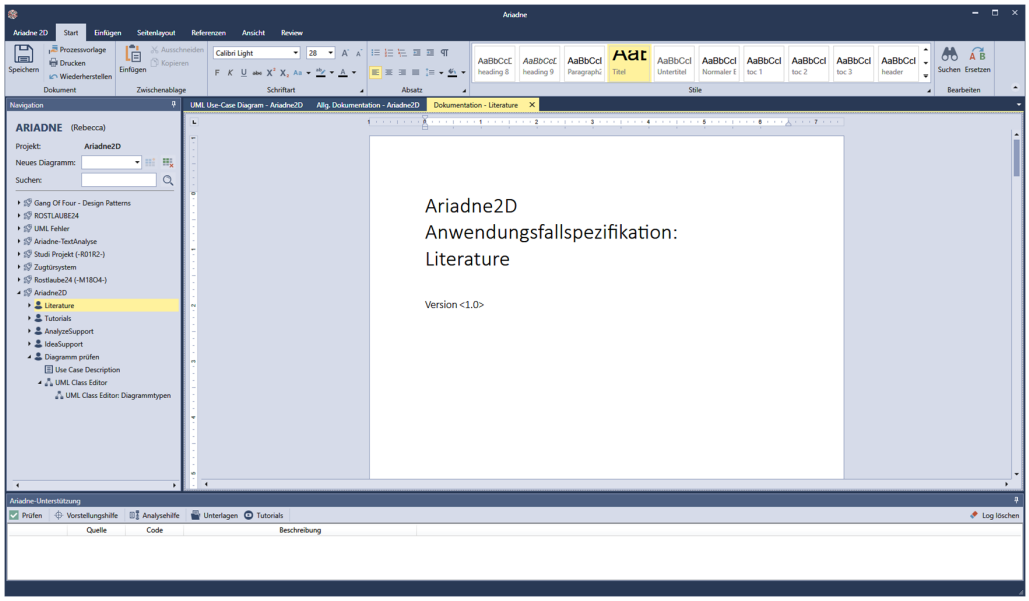


Abbildung 6.4: Texteditor

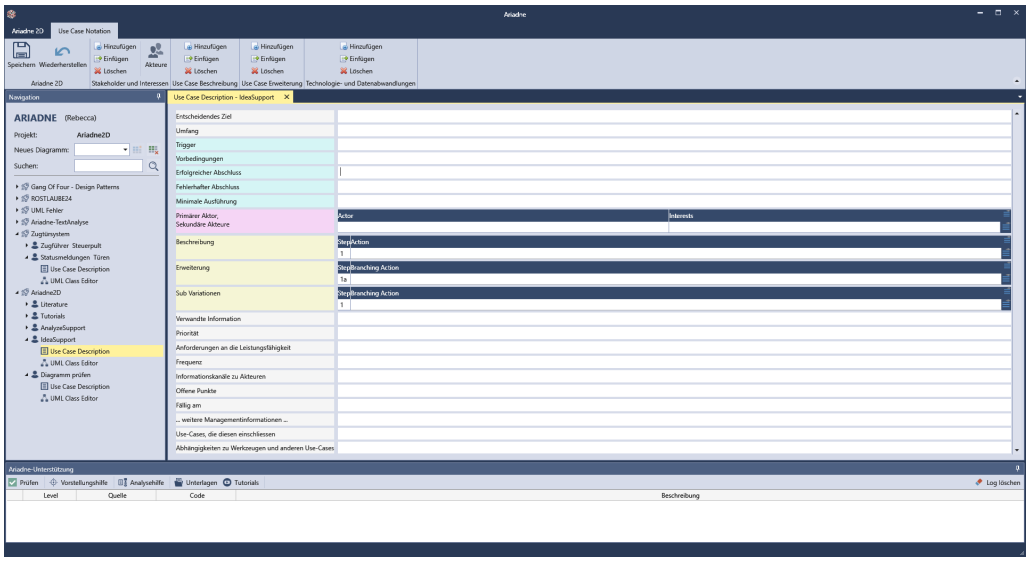


Abbildung 6.5: Template zur Erfassung strukturierter Use-Case-Beschreibungen

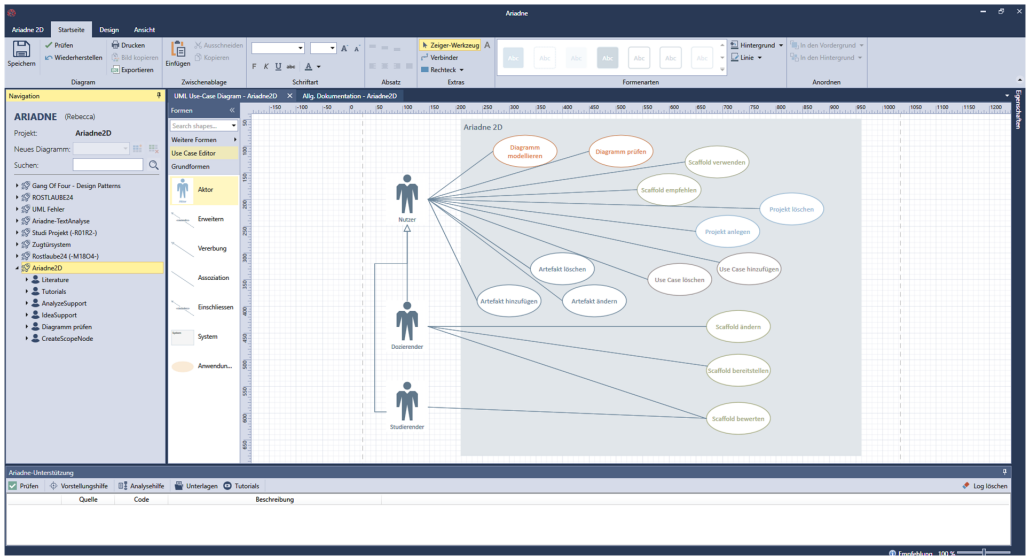


Abbildung 6.6: Editor zur Modellierung von Use-Case-Diagrammen

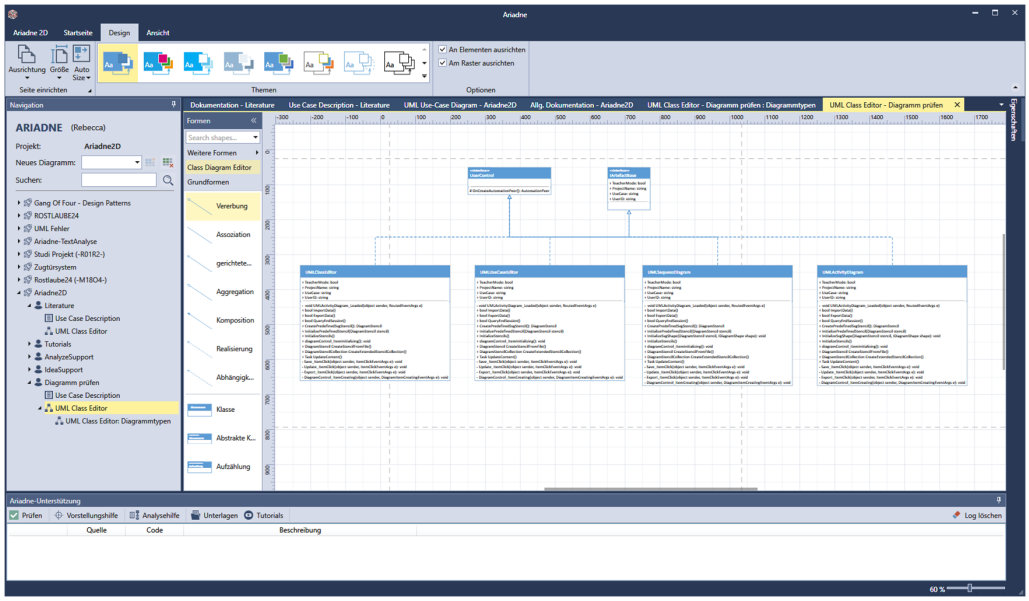


Abbildung 6.7: Editor zur Modellierung von Klassendiagrammen

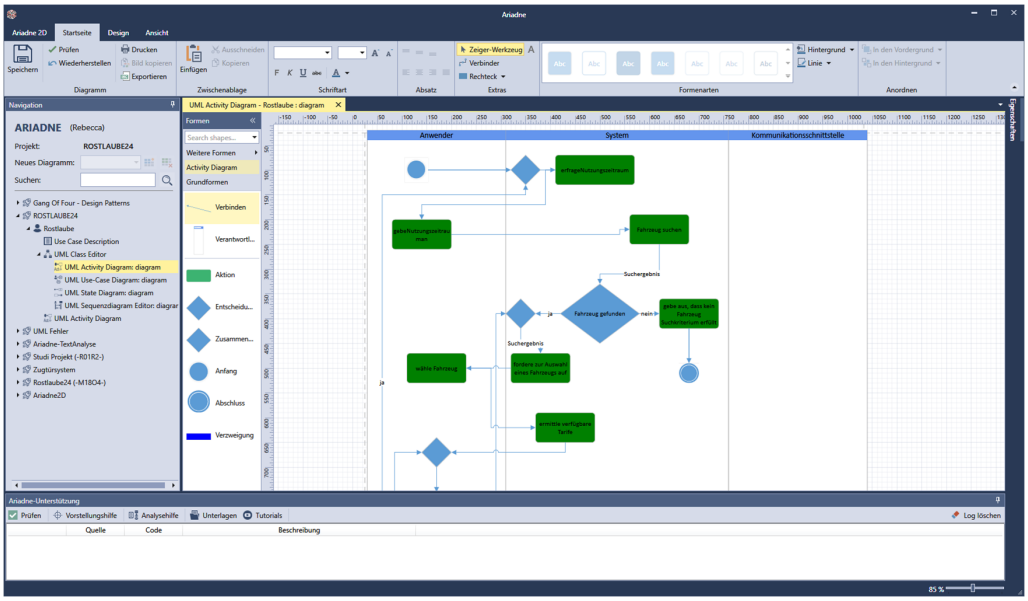


Abbildung 6.8: Editor zur Modellierung von Aktivitätsdiagrammen

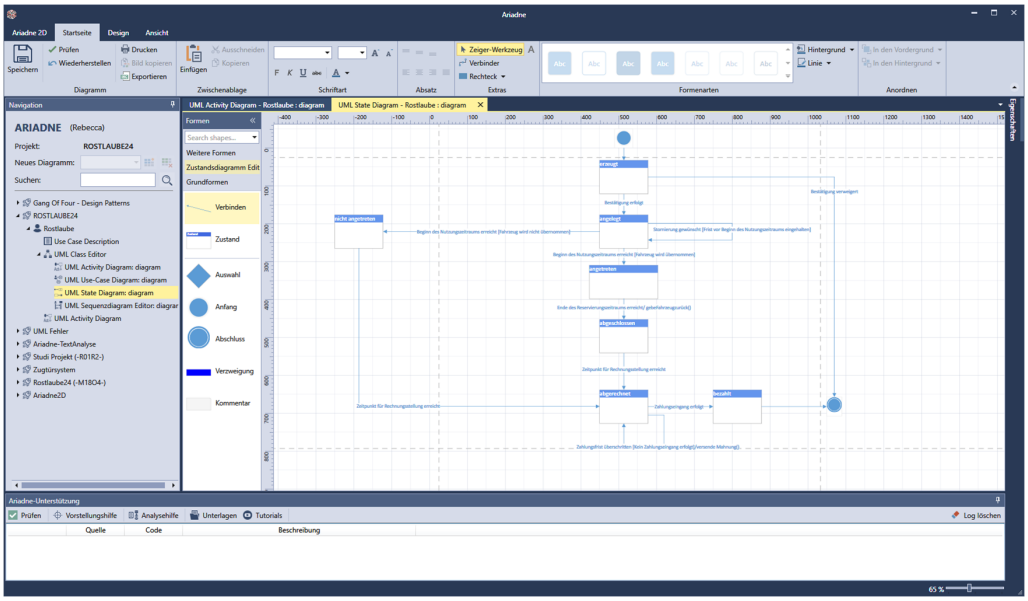


Abbildung 6.9: Editor zur Modellierung von Zustandsautomaten

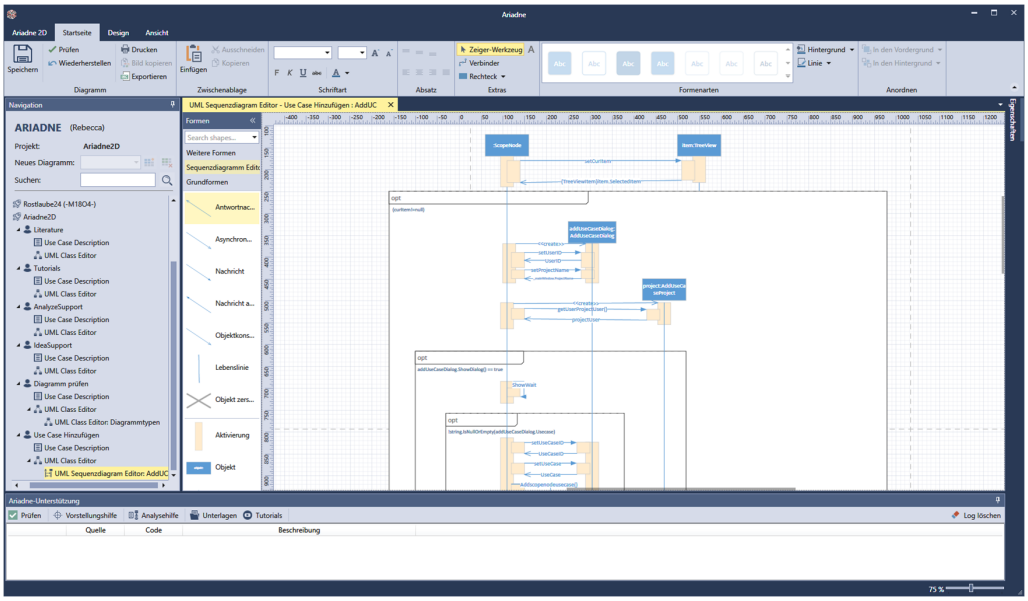


Abbildung 6.10: Editor zur Modellierung von Sequenzdiagrammen

6.3 KONZEPTION DES SCAFFOLDS: UNTERLAGEN

Neben der „anderen“ Modellierungsumgebung, gaben die Studierenden folgende Wünsche an:

- Dokumente
  - Vorlesungsunterlagen
  - Definitionen
  - Spickzettel („Cheatsheets“)
- Beispiele
  - Einfach
  - Komplex
  - Besondere Fälle

Dieser Cluster der Fragebogenstudie wurde als Scaffold „Unterlagen“ umgesetzt. Diese Unterstützungsmaßnahme soll es Nutzern ermöglichen, ohne Verlassen der Software definierte Unterlagen im Rahmen der Software zu verwenden. Nach der vorgestellten CLT bzw. CTML (Abschnitt 3.2.1 und Abschnitt 3.4) gilt es Materialien und Aktivitäten so zu gestalten, dass dadurch die verursachte Belastung (Intrinsic und Extraneous Cognitive Load, also inhaltliche und äußeren kognitive Belastung) reduziert wird. Somit können Lernende die freien kognitiven

Kapazitäten nutzen, Lernaktivitäten werden angeregt und die ungenutzten Kapazitäten können zudem zur Konstruktion von Schemata genutzt werden (Germane Cognitive Load; Schmidt-Weigand et al., 2008; Sweller, Van Merrienboer & Paas, 1998).

Linn (1992), Hislop und Ellis (2009) schlagen zudem die Arbeit mit Templates vor (Abschnitt 6.13). Dieser Vorschlag wurde ebenfalls umgesetzt und Prozessvorlagen nach Kruchten (2004) für die Erfassung von Requirements werden angeboten.

Speziell zu den Beispielen fordern Studierende häufig „Lösungsbeispiele“. Wissenschaftlich betrachtet, setzen sich „Lösungsbeispiele“ zusammen aus der Problemstellung [oder Problemformulierung], Lösungsschritten und der endgültigen Lösung der Aufgabe"(Schmidt-Weigand et al., 2008, S.367; siehe auch Arnold, Kremer & Mayer, 2017; Renkl, 2005). Lösungsbeispiele sind auch als *worked examples* bekannt und die Vorgänger von sogenannten *gestuften Lernhilfen* (Franke-Braun, 2004; siehe auch Arnold et al., 2017). Nach Schmidt-Weigand et al. (2008) wurden bislang für Aufgaben mit eindeutigem Lösungsweg gestufte Lernhilfen untersucht (Forschergruppe Kassel, 2006; Franke-Braun, 2004). Schmidt-Weigand et al. (2008) leitet auch ab, dass bei Aufgaben, die mehrere Lösungen zulassen, was auch auf den Lernkontext dieser Arbeit zutrifft, „Hilfen formuliert werden müssen, die die Wahl des Lösungsweges offen lassen. Die Hilfen dienen in dem Fall dazu, zunächst eine Repräsentation der Aufgabe aufzubauen und Lösungswege selbstständig zu suchen und zu verfolgen“ (Schmidt-Weigand et al., 2008, S. 381).

Durch das Angebot von Hilfsmitteln, die prinzipiell ein didaktisches Setting unter der Verwendung von Scaffolding ermöglichen, bietet die Software Dozierenden die Möglichkeit verschiedene Arten von Beispielen hinzuzufügen.

Unter Einbezug der aus den Schwierigkeiten abgeleiteten Maßnahmen und den obigen Ausführungen, können folgende High-Level-Anforderungen an eine Modellierungsumgebung definiert werden:

Für die prototypische Implementierung zur Nutzung von Unterlagen in der IDE wurden konkret folgende Designentscheidungen getroffen:

- Die Anzeige der Unterlagen wird angelehnt an die Verwendung gängiger Dokument- Viewer.
- Nutzer haben die Möglichkeit, rollenbasiert Unterlagen zu erstellen und zu verändern.
- Die Unterlagen werden in einer Datenbank abgelegt.
- Die Unterlagen können in den Formaten Pdf, Docx und Rtf abgelegt werden.
- Die Struktur der Unterlagen muss Vorlesung, Unterlagen, Zusatzinformationen und Beispiele beinhalten.

Abbildung 6.11 zeigt den allgemeinen Aufbau der Oberfläche, die die verschiedenen Unterlagen bereitstellt.

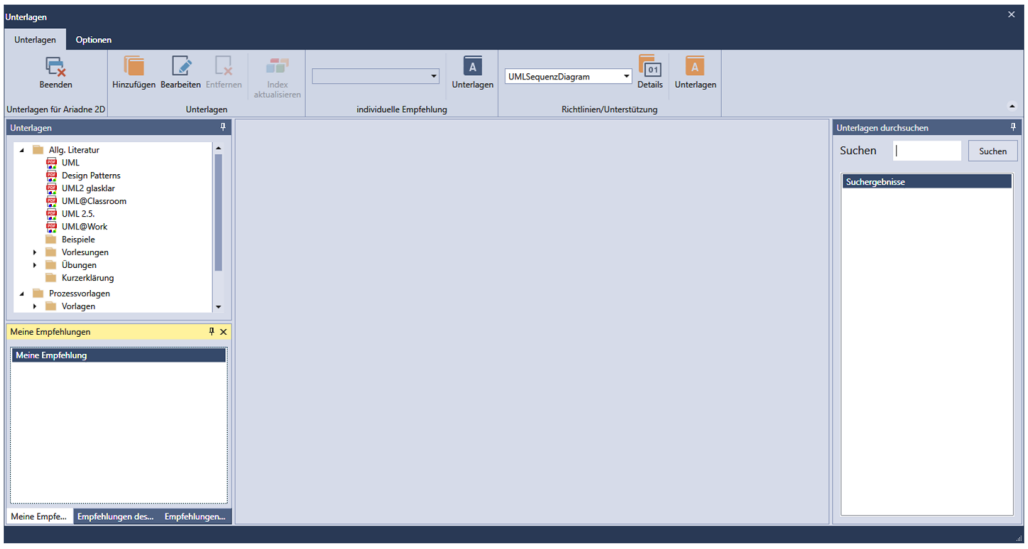


Abbildung 6.11: Der Aufbau des Scaffolds Unterlagen

6.3.1 *Struktur der Unterlagen*

Basierend auf den obigen Erläuterungen wurde für die Unterlagen folgende Struktur gewählt:

- Allgemeine Literatur
  - Beispiele
  - Vorlesungen
  - Übungen
  - Kurzerklärungen

Diesen Abschnitt dürfen ausschließlich Nutzer, die als Dozierende berechtigt sind, verändern. Es gibt die Kategorie der Beispiele, Vorlesungsunterlagen, Übungsunterlagen und sogenannte Kurzerklärungen. Letztere bilden den Wunsch der Studierenden nach „Spickzetteln bzw. Cheatsheets“ ab.

Ein Abschnitt „Prozessvorlagen“ wird eingeführt, in dem, wie bereits angedeutet, Templates zur Verfügung gestellt werden können. Dieser Abschnitt darf ausschließlich von Dozierenden verändert werden.

Außerdem gibt es noch den Abschnitt „Dokumentation“, der von jedem Benutzer verändert werden darf. Dieser dient dazu, Zusatzinformationen zu einer Aufgabe, bzw. einer Problemstellung zu sammeln. Diese Art von

Unterlagen ist abhängig vom Projekt (d. h. der obersten Ebene, die zu Beginn der Nutzung definiert wurde, siehe Abb. 6.12).

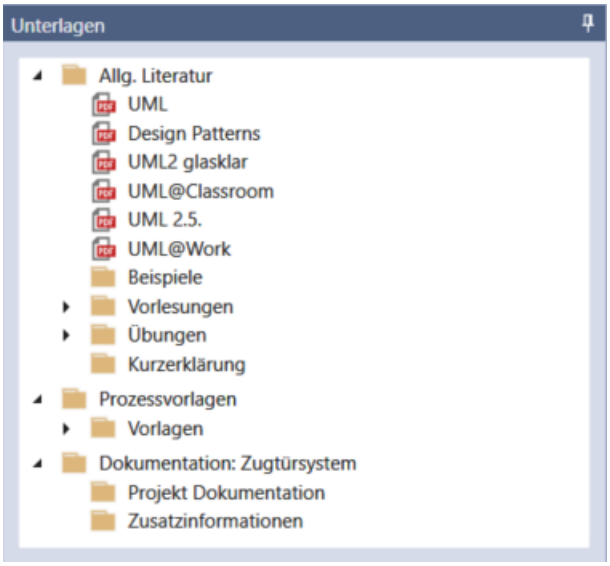


Abbildung 6.12: Die Struktur der Unterlagen, aufgebaut nach den Ergebnissen der Fragebogenstudie

6.3.2 *Durchsuchbare Dokumente*

Wie in diesem Abschnitt bereits beschrieben, orientiert sich diese Arbeit in der Konzeption der Scaffolds an der CLT. Das bedeutet, dass darauf geachtet wurde Intrinsic und Extraneous Cognitive Load, also inhaltliche und äußere kognitive Belastung möglichst gering zu halten. Deshalb sind die Unterlagen, die dort für Studierende ausgewählt worden sind, durchsuchbar (Abb. 6.13). Die Suche ist als Volltextsuche aufgebaut. Die Durchsuchbarkeit wird durch einen Index gewährleistet.

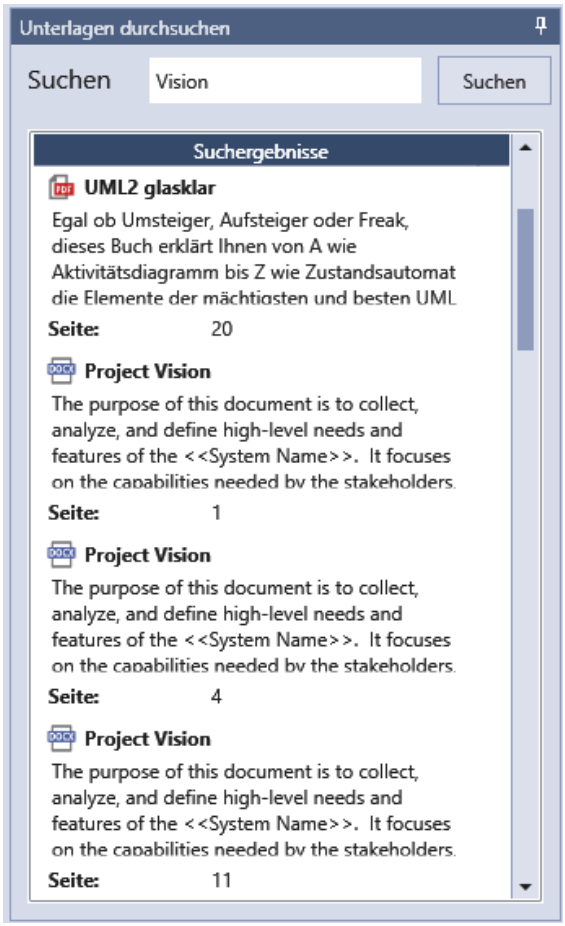


Abbildung 6.13: Die Unterlagen sind durchsuchbar

6.3.3 *Rating von Dokumenten*

Ein Ziel dieses Dissertationsvorhabens ist es eine konstruktivistische Lernumgebung bereitzustellen, die studierendenzentriertes Lehren und Lernen ermöglicht. Dazu wird es als Notwendigkeit betrachtet, neben der Möglichkeit zur Bereitstellung von verschiedenen Unterstützungsmaßnahmen, für Studierende und Lehrende die Möglichkeit zu bieten, diese auch evaluieren, d. h. bewerten zu können. Dies wurde folgendermaßen umgesetzt:

- Ein Dozierender hat die Möglichkeit Studierenden Unterlagen mittels eines „Dozententipps“ zu empfehlen. Dies ist optional.
- Zudem haben Nutzer die Möglichkeit, einzelne Seiten in einem Dokument zu empfehlen.
- Studierende können mittels eines 5-Sterne-Ratings untereinander einzelne Unterlagen empfehlen.

Abbildung 6.14 zeigt exemplarisch die Anzeige bewerteter Unterlagen für einen Nutzer.

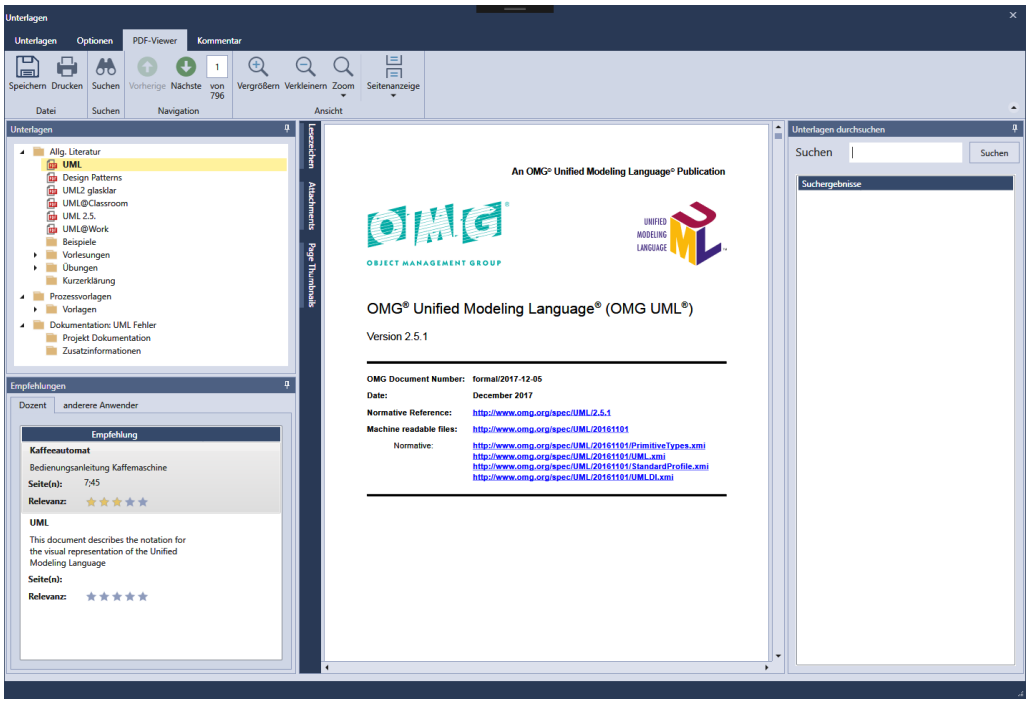


Abbildung 6.14: Exemplarische Darstellung bewerteter Unterlagen

6.4 KONZEPTION DES SCAFFOLDS: TUTORIALS

Neben Unterlagen, gaben die Studierenden folgende Wünsche an:

- Tutorials
- Internet

Dieser Cluster der Fragebogenstudie wurde als Scaffold „Tutorials“ umgesetzt. Diese Unterstützungsmaßnahme soll es Nutzern ermöglichen, ohne Verlassen der Software definierte Tutorials und aber vor allem Online-Material im Rahmen der Software zu nutzen. Unter Einbezug der aus den Schwierigkeiten abgeleiteten Maßnahmen und den obigen Ausführungen, können folgende High- Level- Anforderungen an eine Modellierungsumgebung definiert werden:

- Die Software muss eine Unterstützungsmöglichkeit zur Nutzung von Online-Material bereitstellen.
- Die Software muss die Möglichkeit bieten geeignetes Online Material strukturiert anzuzeigen.

- Die Software sollte verschiedene Rollen mit differenzierten Rechten anbieten, die die Möglichkeit zum Hinzufügen und Löschen eines Tutorials haben.
- Die Software sollte die Möglichkeit bieten die Tutorials zu bewerten.

Für die prototypische Implementierung zur Nutzung von Tutorials in der IDE wurden konkret folgende Designentscheidungen getroffen:

- Die Anzeige von Tutorials wird mittels eines Browser-Plugins realisiert. Mit Hilfe dessen ist es auch möglich Video- und Audiomaterial abzuspielen.
- Nutzer haben die Möglichkeit, rollenbasiert Unterlagen zu erstellen und zu verändern.
- Aufgrund der Flüchtigkeit des Online Materials und um Eigentümerrechte zu gewährleisten, wird hier die URL und eine Kurzbeschreibung in der Datenbank persistiert.

Abbildung 6.15 zeigt den Aufbau der Oberfläche zur Nutzung von Tutorials, Abb. 6.16 zeigt die Möglichkeit Tutorials hinzuzufügen.

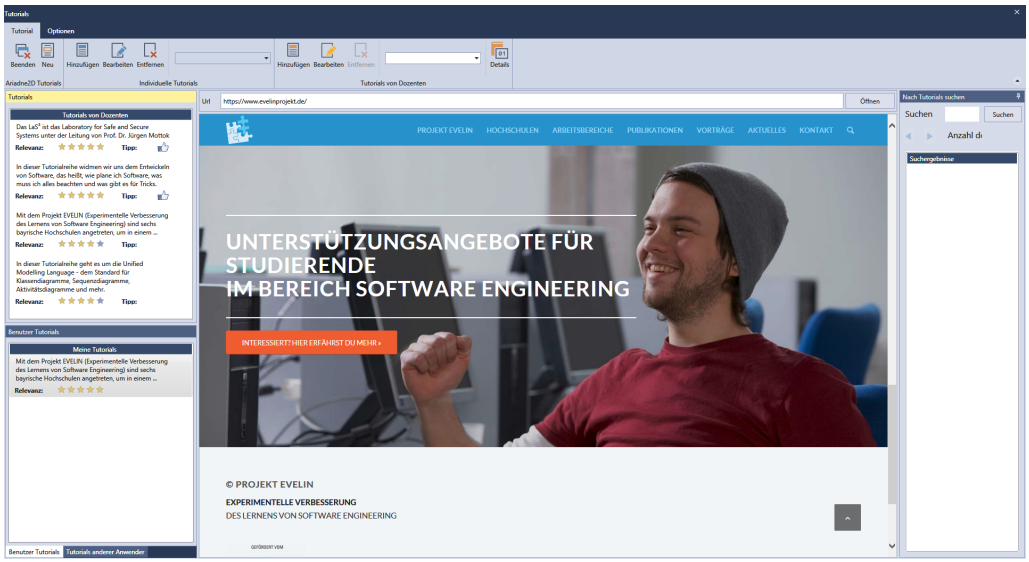


Abbildung 6.15: Der Aufbau des Scaffolds Tutorials

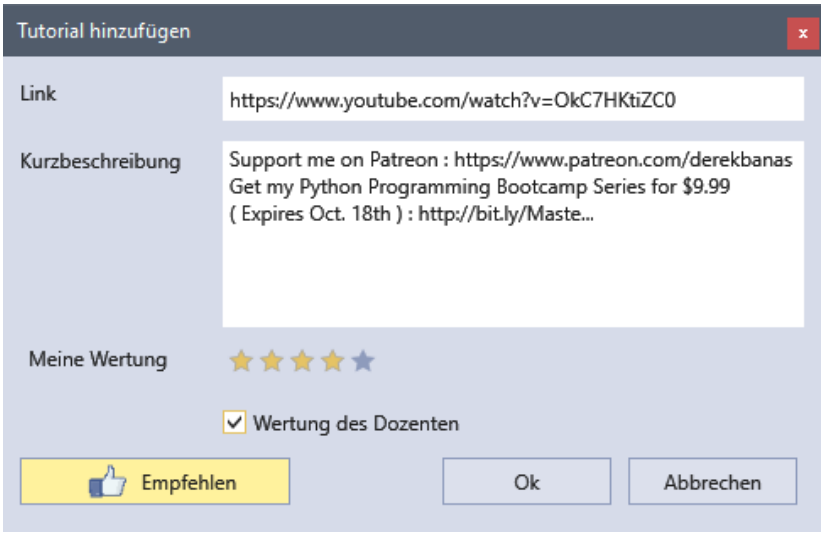


Abbildung 6.16: Dozierende wie auch Studierende können Tutorials hinzufügen

Die beiden Scaffolds Unterlagen und Tutorials weisen sicherlich Gemeinsamkeiten auf. Allerdings wurde die Differenzierung zwischen diesen beiden Scaffolds bewusst getroffen:

- ... um Offline- Material klar von Online-Material zu trennen.
- ... um Audio- und Videomaterial von Dokumenten zu trennen. (Auch wenn mittels des Browser-Plugins die Betrachtung weiteren Online-Materials möglich ist.)
- ... um eine Möglichkeit zu schaffen geschütztes Material zur Verfügung stellen zu können.

Die Empfehlung bzw. Bewertung des Online-Materials ist aus Sicht des Autors gerade bei diesem Scaffold von zentraler Bedeutung, da viele Studierende die Schwierigkeit haben, geeignetes Material selbst zu identifizieren. Hier sei auch der Verweis gegeben, dass die Bereitstellung der Unterlagen wie auch der Tutorials keinen spezifischen Regeln ausgesetzt sind, sodass diese Komponenten der Software allgemein genutzt werden könnten.

6.5 KONZEPTION DES SCAFFOLDS: FEEDBACK

Dieser Abschnitt beschreibt die Überlegungen und Konzeptionen, die aus dem Cluster „Feedback“ abgeleitet wurden:

- Feedback
  - „schnelles kurzes Feedback“

  - Antworten auf die Fragen „Ist das in Ordnung?“ und „eine klare Antwort“

Studierende wünschen sich Rückmeldungen zu ihren erstellten Diagrammen und erkundigen sich deshalb bei Dozierenden nach der Korrektheit der von ihnen erstellten Diagramme. In der Fragebogenstudie äußert sich dies durch Wünsche wie „schnelles kurzes Feedback“, was darauf hindeutet, dass Dozierende zu ausführliche Erläuterungen abgegeben haben oder eben wie die beiden weiteren hier gelisteten Antworten andeuten, unklare, uneindeutige Antworten geliefert haben.

Bei der Modellierung von Softwaresystemen ist die Bewertung von Modellen nicht trivial, da meist mehr als nur eine Lösung zulässig ist (Ali & Terengganu, 2007; Cheers, Javed, Lin & Smith, 2019; Gelhausen, Landh & Sven, 2008; Reischmann & Kuchen, 2019). Gelhausen et al. (2008) geht soweit und behauptet „an assessment based on a sample solution is problematic“ und regt weiterhin an, studentische Lösungen nicht gegen eine „Musterlösung“ zu vergleichen, sondern jede Lösung individuell zu prüfen. Bezogen auf die Konzeption von Hilfestellungen trifft die von Schmidt-Weigand et al. (2008) formulierte Notwendigkeit der Formulierung von Hilfen, die die Wahl des Lösungsweges offen lassen, zu.

Deshalb wurde prototypisch eine „Problemerkennung“ integriert, um Studierenden die Möglichkeit zu bieten, einerseits eigenständig und andererseits systematisch immer die gleichen Verstöße gegen Richtlinien der Modellierung (z. B. Booch, Rumbaugh und Jacobson (1999); Balzert (2001)) zu zurückzumelden. Über die Empfehlung können Dozierende zu spezifischen Hilfestellungen raten, Studierende entscheiden aber selbst über den Lösungsweg, den sie wählen:

- Die Software muss die Möglichkeit bereitstellen, vordefinierte Richtlinien in einem Diagramm zu erkennen.
- Die Software muss die Möglichkeit bereitstellen, Verstöße gegen definierte Richtlinien in einem Diagramm zu erkennen und zu markieren.

Für die prototypische Implementierung zur Nutzung des Feedbacks in der IDE wurden konkret folgende Designentscheidungen getroffen:

- Nutzer können über den Button „Prüfen“ das erstellte Diagramm prüfen lassen.
- Erkannt werden problematische Stellen, die gegen vorab definierte Richtlinien verstoßen. Diese Richtlinien wurden auf Basis des Problemkatalogs vordefiniert, sind aber für Dozierende veränderbar und erweiterbar.

- Die problematischen Stellen werden rot markiert und zusätzlich im unteren Bereich der Oberfläche gelistet.
- Ein Element der Liste enthält eine ID und die Beschreibung des Verstoßes.
- Ein Klick auf ein Element der Liste markiert die betroffenen Elemente im Diagramm und zeigt über ein Pop-up Fenster die verletzte Richtlinie und bietet über die Buttons oberhalb der Beschreibungen Empfehlungen welches Scaffold dabei unterstützen könnte.
- Die spezifischen Empfehlungen pro Richtlinie sind von Dozierenden konfigurierbar. Für jede Richtlinie kann ein Dozent Empfehlungen abgeben bzw. ändern.

Abbildungen 6.17 bis 6.18 zeigen zwei Wege, das erstellte Diagramm „prüfen“ zu lassen. Beide Optionen erfüllen die gleiche Funktion.

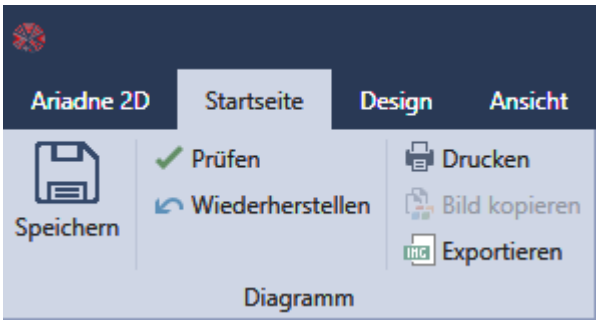


Abbildung 6.17: Der Nutzer kann das erstellte Diagramm „prüfen“ lassen. (Variante 1)

Abbildung 6.18: Der Nutzer kann das erstellte Diagramm „prüfen“ lassen. (Variante 2)

Dozierende können Unterlagen empfehlen. Über die Liste auf der linken Seite in Abb. 6.19 wird ausgewählt, bei welchem Problem die Unterlagen empfohlen werden, über die Liste auf der rechten Seite können die entsprechenden Unterlagen und die entsprechenden Seiten ausgewählt werden sowie eine Wertung der Quelle vorgenommen werden.

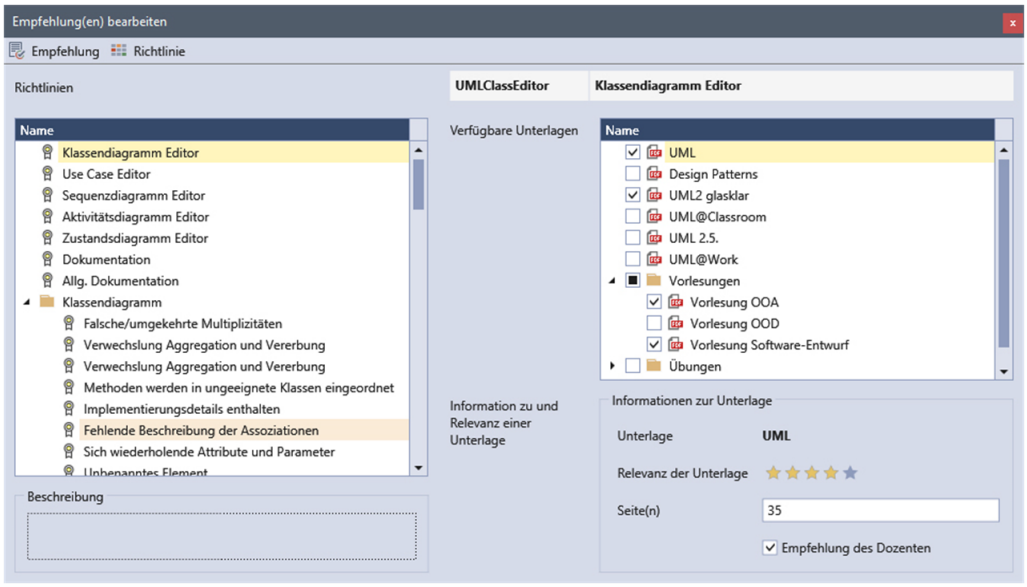


Abbildung 6.19: Konfiguration von Empfehlungen

6.6 KONZEPTION DES SCAFFOLDS: VERSIONIERUNG

Die Versionierung von erstellten Artefakten ist kein Scaffold, der aus der Fragebogenstudie abgeleitet wurde, sondern eine Hilfestellung, die inspiriert von der Arbeit von Thomasson et al. (2006a), konzipiert wurde. Durch die Versionierung der Artefakte ist es Nutzern möglich, einerseits zu einer älteren Version des Modells zurückzukehren, andererseits können Dozierende auch den Entstehungsprozess beispielsweise gemeinsam mit Studierenden diskutieren.

Deshalb wurden folgende Anforderungen an eine Software abgeleitet:

- Die Software muss Möglichkeit bereitstellen, Artefakte zu versionieren.
- Die Software muss Möglichkeit bereitstellen, versionierte Artefakte wiederherzustellen.

Dazu wurden folgende Designentscheidungen getroffen:

- Artefakte werden nach jeder Speicherung mit entsprechenden Parametern, wie beispielsweise den exakten Zeitstempeln, in einer Datenbankstruktur abgelegt.
- Zur Wiederherstellung des Artefakts erhält der Nutzer eine Eingabeaufforderung und kann zwischen den abgelegten Versionen seines Artefakts über die Wiederherstellung entscheiden.

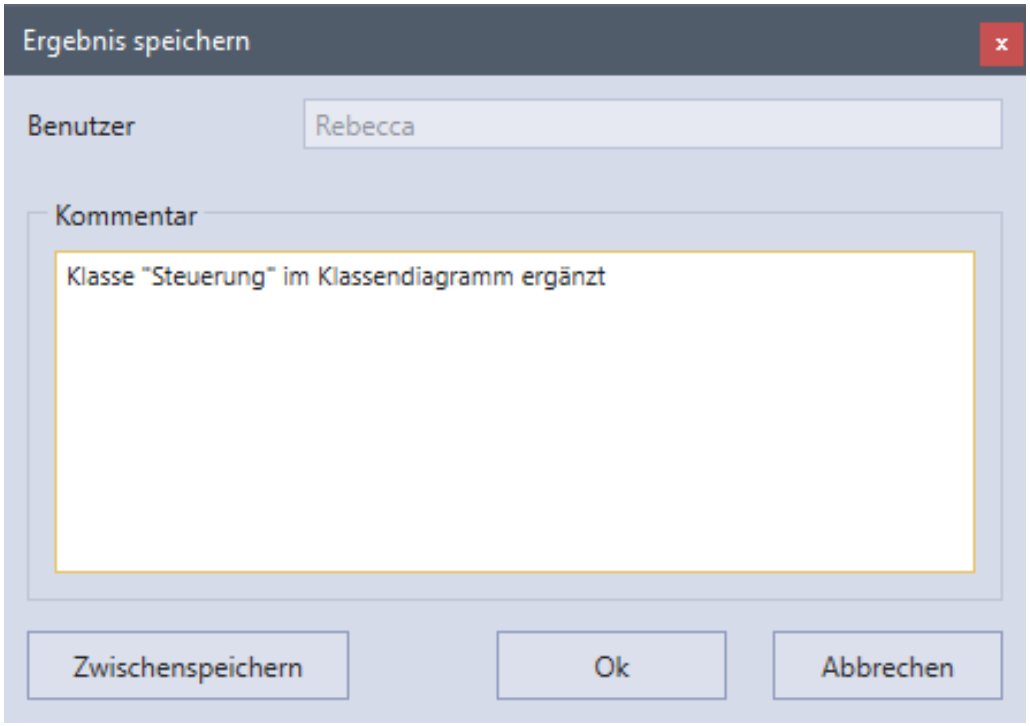


Abbildung 6.20: Versionierung beim Speichern von Artefakten: Dialog zum Speichern des aktuellen Artefakts

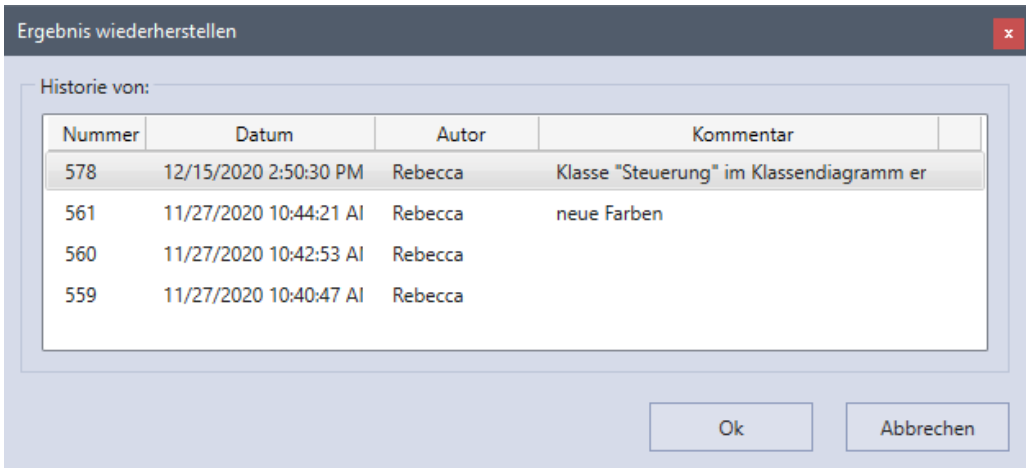


Abbildung 6.21: Versionierung beim Speichern von Artefakten: Wiederherstellen des Artefakts

Abbildungen 6.20 bis 6.21 zeigen die Aufforderungen, die Nutzer erhalten, wenn ein Artefakt gespeichert wird bzw. wenn Nutzer ein Artefakt wiederherstellen möchten.

6.7 KONZEPTION DES SCAFFOLDS: KOMMENTARE

Neben der Versionierung von Artefakten finden sich auch Kommentare zur Unterstützung für Studierende in der Literatur (Linn, 1992; Linn & Clancy, 1992). Die Kommentierung von Programmcode, aber auch anderen Artefakten soll Studierende zum Beispiel unterstützen, Entscheidungen nachvollziehbarer zu gestalten (Linn, 1992; Linn & Clancy, 1992). Auch im Zusammenhang mit der Versionierung in Abschnitt 6.6 wird die Kommentierung als Unterstützung genannt. Studierende sollen ihre Version mit Kommentaren versehen (Thomasson et al., 2006a). Dieser Ansatz wurde für Arbeit aufgegriffen und folgende Anforderungen an eine Software abgeleitet:

- Die Software muss Möglichkeit bereitstellen bei jeder Speicherung einen Kommentar für die Version/den Fortschritt zu hinterlegen.
- Die Software muss Möglichkeit bereitstellen Unterlagen zu kommentieren.
- Die Software muss Möglichkeit bereitstellen Artefakte zu kommentieren.

Dazu wurden folgende Designentscheidungen getroffen:

- Beim Speichern eines Artefakts werden Nutzer aufgefordert einen Kommentar abzugeben.
- Die Kommentare sind für die jeweiligen berechtigen Nutzer für das Projekt sichtbar.
- Kommentare in den Unterlagen sind für alle Nutzer sichtbar.

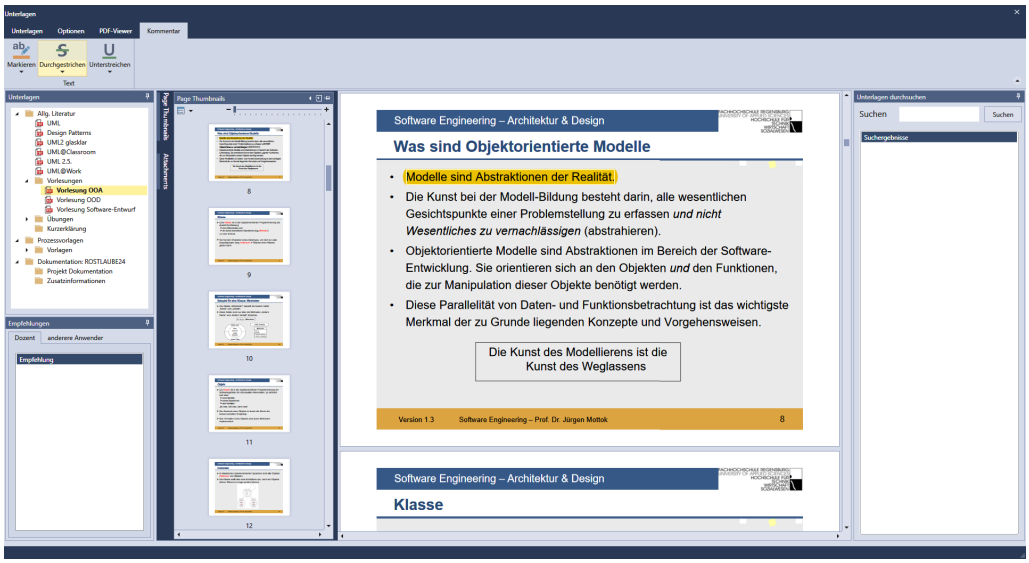


Abbildung 6.22: Markieren von Dokumenten

Innerhalb des Dialogs, der die Unterlagen bereitstellt, ist es möglich Zeilen der Dokumente zu markieren (Abb. 6.22).

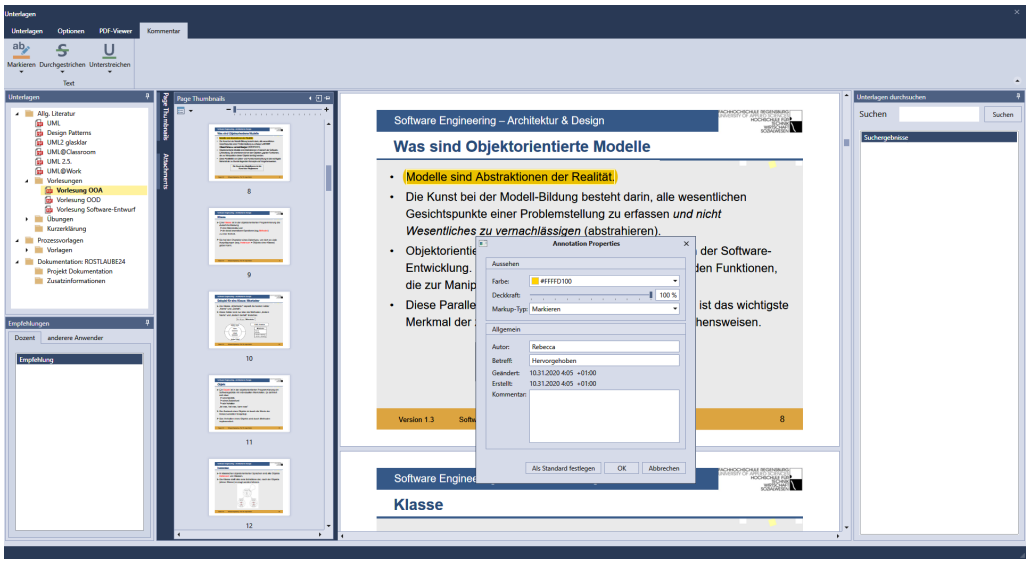


Abbildung 6.23: Kommentarfunktion für Unterlagen

Die Markierungen können über einen Rechtsklick mit einem Kommentar versehen werden (Abb. 6.23).

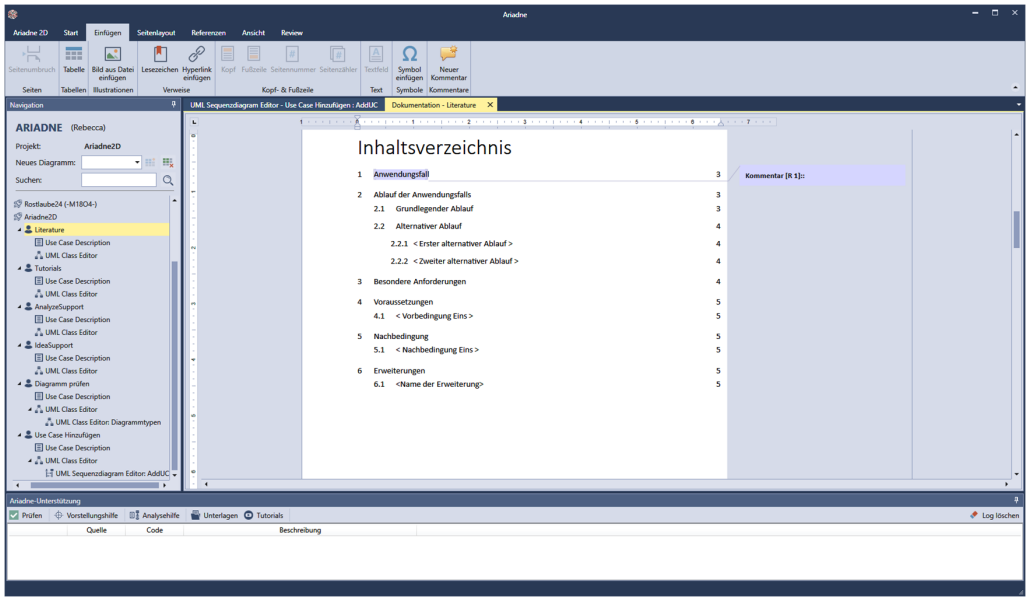


Abbildung 6.24: Kommentarfunktion im Texteditor

Beim Speichern eines Artefakts werden Nutzer aufgefordert einen Kommentar abzugeben (Abb. 6.24).

6.8 Zusammenarbeit und Hilfe von Gruppenmitgliedern

Kollaborationsmöglichkeiten zwischen Studierenden wurden in Publikationen als Scaffolds erfolgreich evaluiert (Chalk, 2000; Manno, Palmieri & Scarano, 2011). Der Wunsch nach Zusammenarbeit und Hilfe zwischen Gruppenmitgliedern wurde auch in der Fragebogenstudie erhoben. Für die Modellierungsumgebung wurde deshalb die Anforderung aufgestellt, dass diese die Möglichkeit bieten sollte, Berechtigungen für ein Projekt zu vergeben. Über die Datenbank als zentrale Komponente ist es möglich, dass berechtige Nutzer die Artefakte eines Projekts anlegen und verändern können. Über die Export-/Import-Funktion ist es zusätzlich möglich Artefakte auszutauschen. Eine weitere Möglichkeit ist im Rahmen zukünftiger Arbeiten die Integration eines Chatprogramms in das Softwaresystem. Dies ist aber in der vorliegenden Arbeit nicht vorgesehen.

6.9 Konzeption des Scaffolds: Vorschläge - Analysehilfen

Dieser Abschnitt beschreibt die Überlegungen und Konzeptionen, die aus dem Cluster „Vorschläge“ abgeleitet wurden:

- Vorschläge
- Musterlösungen

Die beiden Cluster „Vorschläge“ und „Musterlösungen“ wurden ebenso wie Beispiele oder Feedback von Studierenden gefordert. Wie bereits in Abschnitt 6.3 im Rahmen der Lösungsbeispiele beschrieben, lässt die Modellierung bzw. ein Softwareentwurf mehrere Lösungen zu (Ali & Terengganu, 2007; Cheers et al., 2019; Gelhausen et al., 2008; Reischmann & Kuchen, 2019). Entwerfen entspricht in der Softwareentwicklung der Beantwortung der Fragestellung nach der Lösung des Problems, wie eine Software gebaut werden soll. Ein Entwurf ist dabei nichts anderes als eine Möglichkeit aus dem Lösungsraum einer Softwareentwicklung. Es gibt also viele Möglichkeiten der Umsetzung, die unterschiedliche Eigenschaften im Hinblick auf den Aufwand für diese und auf die verschiedenen resultierenden Qualitätsmerkmale haben (Sommerville, 2013, S. 33; Prechelt, 2005, S. 5). „Es ist schwierig, unter diesen Möglichkeiten die brauchbaren [Lösungen] zu erkennen und die günstigste aus diesen auszuwählen“ (Prechelt, 2005, S. 5). Diese Eigenschaften eines Softwareentwurfs lassen also keine „Musterlösung“ für eine Aufgabe zu. Das ist für Studierende aus Ingenieur-Studiengängen häufig zunächst nicht nachvollziehbar. Deshalb wurde als weiterer Scaffold eine Sammlung existierender Projekte (Abschnitt 6.9.5; s.u.) eingeführt.

Die bislang vorgestellten Unterstützungsmaßnahmen adressieren Probleme, die didaktischen oder ressourcenbezogenen Lernhindernissen zugeordnet werden können. Weitere zentrale Probleme wurden auch für die ersten beiden Schritte des Makroprozesses, der z. B. von Balzert und Balzert (2009) für die objektorientierte Analyse vorgestellt wurden (Abschnitt 4.3) identifiziert und finden sich im Katalog über die identifizierten Schwierigkeiten in Abschnitt 5.3.5:

- Die Bestimmung geeigneter Use-Cases
- Die Bestimmung geeigneter Klassen

Deshalb werden für diese beiden Phasen zwei Unterstützungsmaßnahmen auf Basis existierender Techniken (wie beispielsweise die Identifikation von Klassenkandidaten mittels der Abbott Technik) eingeführt. Studierende sollen bei diesen Problemen erste Ansätze erhalten und so an die Anwendung der Techniken herangeführt werden.

Im Sinne des Designs konstruktivistischer Lernumgebungen und dem damit einhergehenden Instruktionsdesigns, ist es dediziertes Ziel, geeignete Unterstützungsmaßnahmen (Scaffolding)[28] zur Bearbeitung von Aufgaben bereitzustellen Abschnitt 3.2.1. Es geht also darum, Studierenden erste Ansätze zur Identifikation (von Use-Cases und Klassen) aufzuzeigen und nicht darum, die Lösung „per Knopfdruck“ zu präsentieren.

Eine Identifikation von Use-Cases geschieht auf Basis von Anforderungsdokumenten. Nach Rupp und die SOPHISTen (2014) wird im agilen Requirements Engineering „in der initialen Phase des Projekts das zu erstellende Produkt oder System nicht vollständig spezifiziert. Ziel ist es, ein gemeinsames Verständnis des Systems und der Zielsetzung des Projekts zu erzeugen; eine Vision, die nur in groben granularen Anforderungen beschrieben wird. Obwohl immer ein gewisser Vorlauf bei der Erstellung des Lastenhefts vonnöten ist, entwickeln sich das Product-Backlog und das Pflichtenheft parallel zum System“ (Rupp & die SOPHISTen, 2014, S. 508). Das Hilfsmittel der Analyse von Use-Cases basiert auf dem Vision-Dokument, das z. B. im Rational Unified Process vorgeschlagen wird (Kruchten, 2004; Rational Software Corporation, n. d.). Es wurden lediglich kleinere Anpassungen und die Übersetzung in die deutsche Sprache vorgenommen. Die Erstellung von Use-Cases ist der erste Schritt der Modellbildung nach dem Makroprozess und damit eine Form der Abstraktion (Balzert & Balzert, 2009, S. 27). Abstraktion gilt als ein wichtiges Prinzip in der Softwaretechnik und „Abstrahieren“ ist deshalb eine zentrale Kompetenz von Software Ingenieuren (Balzert & Balzert, 2009,

28 Hier bedeutend ist die Beschreibung von Jonassen (1999) zum Einsatz kognitiver Hilfsmittel.

S. 26). Diese Kompetenz gilt als eine anspruchsvolle Aufgabe (Balzert & Balzert, 2009, S. 29). Deshalb gilt im Allgemeinen:

> „Intuitiv verständlich für eine Fachabteilung ist etwas dann, wenn es in seiner Struktur nahe an der Struktur der natürlichen Sprache liegt. Dies ist die Sprache in der eine Fachabteilung seine“Modelle"beschreibt. Es gilt also, die Vorteile der natürlichen Sprache, wie allgemeine Verfügbarkeit und auch gewisse Formalisierungsansätze wie Grundbedeutungen für Worte, Stabilität (Goethe versteht man heute noch) und Standardsatzsemantiken wie Subjekt, Prädikat und Objekt zu nutzen, aber die Unschärfen wie Mehrdeutigkeiten von Worten zu vermeiden"(Sneed & Fleischmann, n. d., S. 11; Booch & Eykholt, 1996).

Für Schritt zwei im Makroprozess, die Identifikation von Klassen, kann die Abbott Technik (auch Substantiv-Verb-Analyse, Substantivanalyse oder Hauptwortanalyse genannt) verwendet werden. Sie ist ein bewährtes Mittel um geeignete Substantive als Klassenkandidaten und Verben als mögliche Methoden für ein Klassendiagramm zu identifizieren. Bei der Analyse der Probleme der Studierenden hat sich gezeigt, dass nicht der erste Schritt der Bestimmung der Substantive und Verben problematisch ist, sondern vielmehr die Anwendung der Methode per se. Häufig wird sie scheinbar gar nicht in Betracht gezogen.

Im Rahmen einer Eyetracking Studie mit anschließenden retrospektiven Interviews, die im Rahmen einer betreuten Masterarbeit (Hutzler, 2018) durchgeführt wurde, konnte die Anwendung dieser Technik auch bei Experten nachgewiesen werden. Dabei hat sich auch gezeigt, dass Experten die Technik als eine Heuristik anwenden, d. h. sie sind sich deren Anwendung nicht bewusst (Hauser, Reuter, Gegenfurtner, Gruber & Mottok, 2019; Hutzler, Hauser, Reuter, Mottok & Gruber, 2018).

Die Bereitstellung dieser Technik als Hilfsmittel mit Vorschlägen zu möglichen Klassen soll einerseits dazu beitragen, Studierende kognitiv nicht zu überlasten und ihnen „den ersten Schritt“ abzunehmen und außerdem dazu anregen, diese Methodik auch unabhängig von der gewählten Software einzusetzen[29].

Dieser Ansatz wird ebenfalls verfolgt, um Studierende dabei zu unterstützen, geeignete Use-Cases (strukturiert) zu identifizieren: Features, die im Vision-Dokument des Rational Unified Process (Kruchten, 2004; Rational Software Corporation, n. d.) in natürlicher Sprache beschrieben wurden, werden auf Subjekt, Prädikat und Objekt hin analysiert, sodass

29 Natürlich ist es notwendig die Abbott Technik vorab im Rahmen der Lehrveranstaltung einzuführen, ein Lehrkonzept dazu ist aber nicht im Umfang der Arbeit enthalten.

Vorschläge für die Benennung von Use-Cases bereitgestellt werden können. In diesem Dokument werden ebenfalls Akteure beschrieben, die ebenfalls aus dem Vision-Dokument gelesen werden. In einem zweiten Schritt gilt es dann die Use-Case Beschreibung zu analysieren und vor allem den Ablauf des Use-Cases in einer strukturierten Form, nach Cockburn (2003) zu erfassen. Dazu wird die Beschreibung eines Use-Cases analysiert, dieser in einzelne Sätze zerlegt und Sätze rot markiert, die nicht den Regeln einer Vorgangsbeschreibung entsprechen (Abb. 6.32).

Die beiden Hilfsmittel sind so konzipiert, dass sie, wenn gewünscht, Vorschläge für Use-Cases sowie Klassen bereitstellen, die von Anwendern anschließend selbst ausgewählt werden müssen. Es wird explizit darauf verzichtet, die vollständigen Techniken anzubieten, da sonst die Lernziele, die Bestimmung von Use-Cases auf Basis von Anforderungsdokumenten und die Bestimmung von Klassen auf Basis von Use-Cases, nicht mehr erfüllt werden und Studierende die Technik nicht mehr selbst durchführen.

6.9.1 *Vorschläge zur Identifikation von Use-Cases und Akteuren*

Bezogen auf die von den identifizierten Schwierigkeiten zur Identifikation von Use-Cases können demnach folgende High- Level- Anforderungen an eine Modellierungsumgebung definiert werden:

- Die Software sollte eine Unterstützungsmöglichkeit zur Analyse eines Anforderungsdokuments bieten.
- Die Software sollte die Möglichkeit bieten, relevante Textausschnitte für Use-Cases anzuzeigen.
- Die Software sollte die Möglichkeit bieten, Vorschläge für Akteure anzuzeigen.
- Die Software sollte die Möglichkeit bieten, Vorschläge für die Benennung der Use-Case anzuzeigen.
- Die Software sollte die Möglichkeit bieten, Vorschläge für die Benennung der Use-Case zu übernehmen.
- Die Software sollte die Möglichkeit bieten, die Benennung eines Use-Cases zu ändern.
- Die Software sollte die Möglichkeit bieten, Use-Case-Vorschläge in die Modellierungsebene zu übernehmen.
- Die Software sollte die Möglichkeit bieten, Use-Case-Vorschläge in die Modellierungsebene zu editieren.
- Die Software sollte die Möglichkeit bieten, Use-Cases hinzuzufügen.
- Die Software sollte die Möglichkeit bieten, Use-Cases zu löschen.
- Die Software sollte die Möglichkeit bieten, Namen und Eigenschaften von vorgeschlagenen Akteuren zu ändern.

- Die Software sollte die Möglichkeit bieten, vorgeschlagene Akteure zu löschen.
- Die Software sollte die Möglichkeit bieten, Akteure hinzuzufügen.

Designentscheidungen:

- Um dieses Hilfsmittel nutzen zu können, muss eine Projektbeschreibung nach dem Template aus dem Rational Unified Process (Kruchten, 2004; Rational Software Corporation, n. d.) vorliegen, in dem eine markierte Tabelle mit gewünschten Features gefüllt ist.
- Die Vorschläge für die relevanten Textstellen werden durch Analyse der Tabelle zu Produkteigenschaften realisiert. Diese liefert Beschreibungen zu Features, die von Stakeholdern gewünscht werden.
- Die Funktionalität zur Analyse des Textes wird über die Einbindung des OpenNLP (nähere Informationen dazu in Abschnitt 7.1.4.6 realisiert, mit Hilfe dessen natürlichsprachliche Texte in die Satzbestandteile Subjekt, Prädikat und Objekt zerlegt werden können. Damit wird die Benennung eines Use-Cases vorgeschlagen.
- Vorschläge für Use-Cases und Akteure können mittels Drag-and-Drop auf die Modellierungsfläche gebracht werden. Neben der Drag-and-Drop Funktionalität ist auch eine Kopierfunktion implementiert.
- Die ausgewählten und erstellten Use-Cases und Akteure können zudem direkt in die Navigationsstruktur übernommen werden.

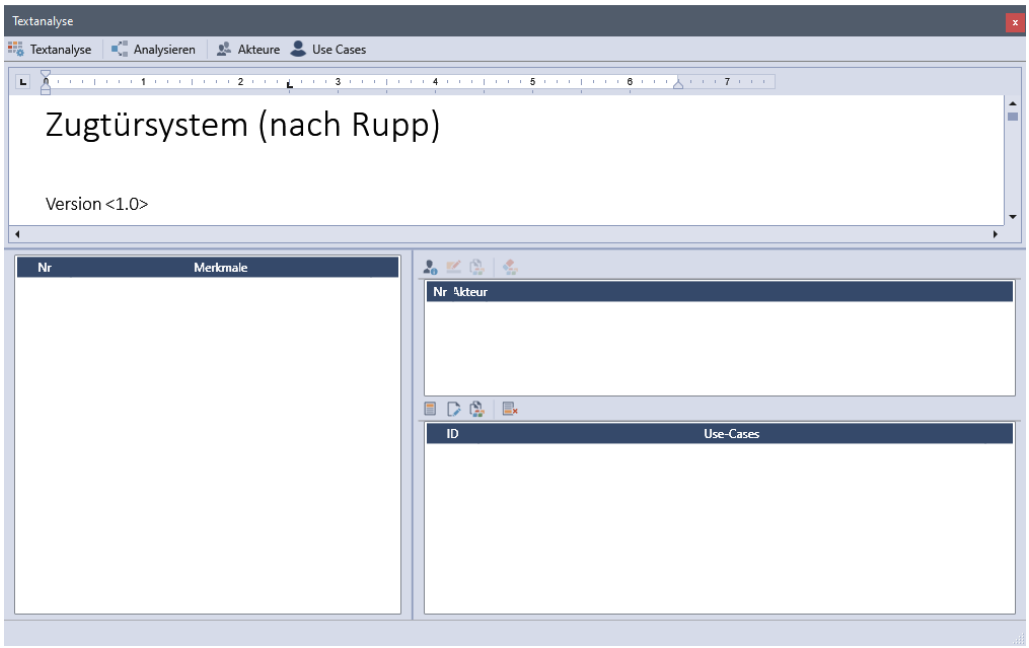


Abbildung 6.25: Dialog zum Scaffold Textanalyse, in der Phase, in der noch keine Analyse durchgeführt wurde

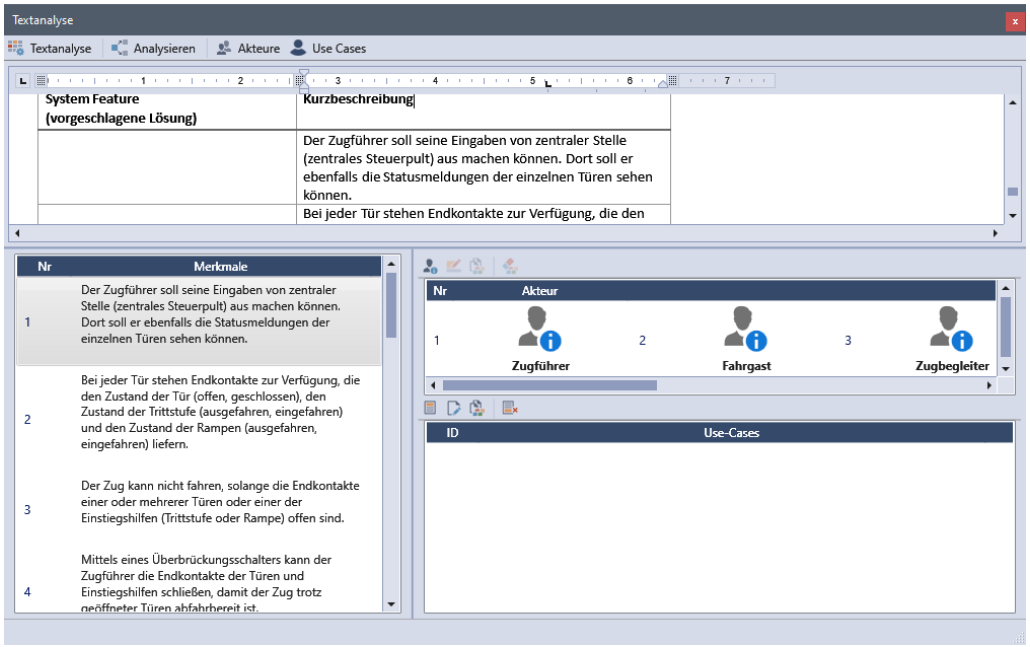


Abbildung 6.26: Dialog nach einem Doppelklick auf eine relevante Textstelle. Dort ist es möglich einen Vorschlag für einen Namen des Use-Cases auszuwählen oder einen eigenen zu definieren. Zudem können für den Use-Case relevante Akteure ausgewählt werden.

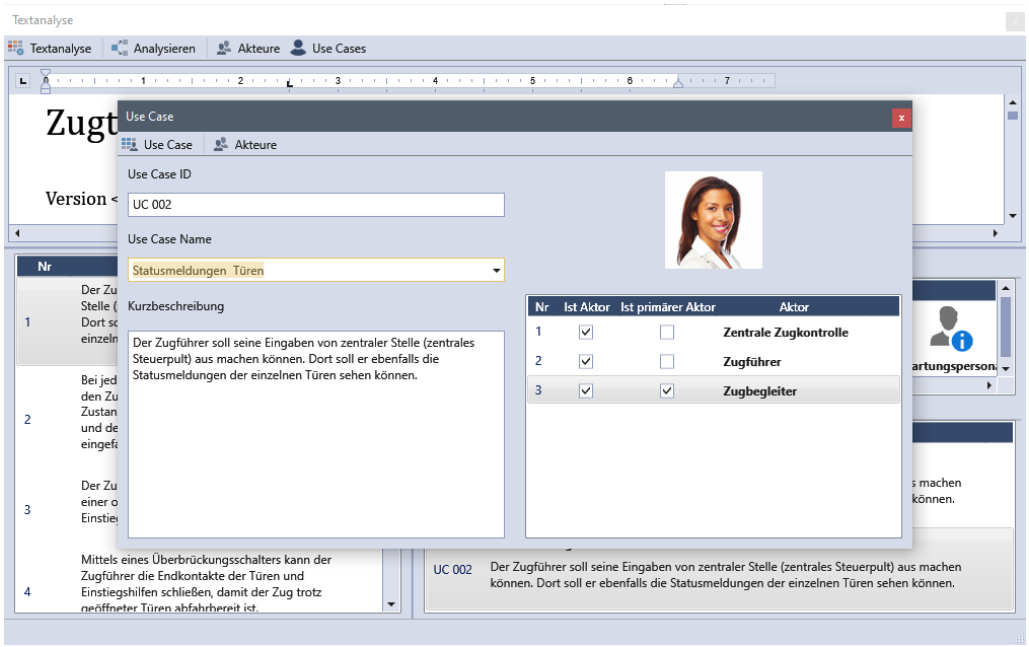


Abbildung 6.27: Dialog nach ausgewählter relevanter Textstelle. Dort ist es möglich einen Vorschlag für einen Namen des Use-Cases auszuwählen oder einen Eigenen zu definieren. Zudem können für den Use-Case relevante Akteure ausgewählt werden.

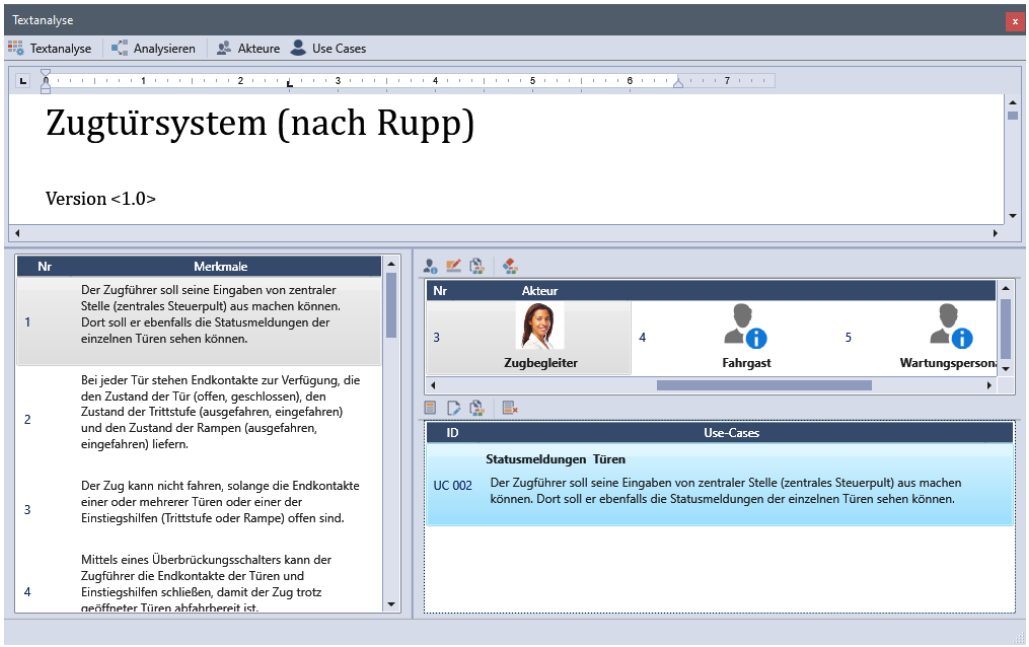


Abbildung 6.28: Dialog des Scaffolds nach der Auswahl eines Use-Cases und des zugehörigen Aktors.

Abbildung 6.29: Dialog zur Änderung der Attribute eines Aktors

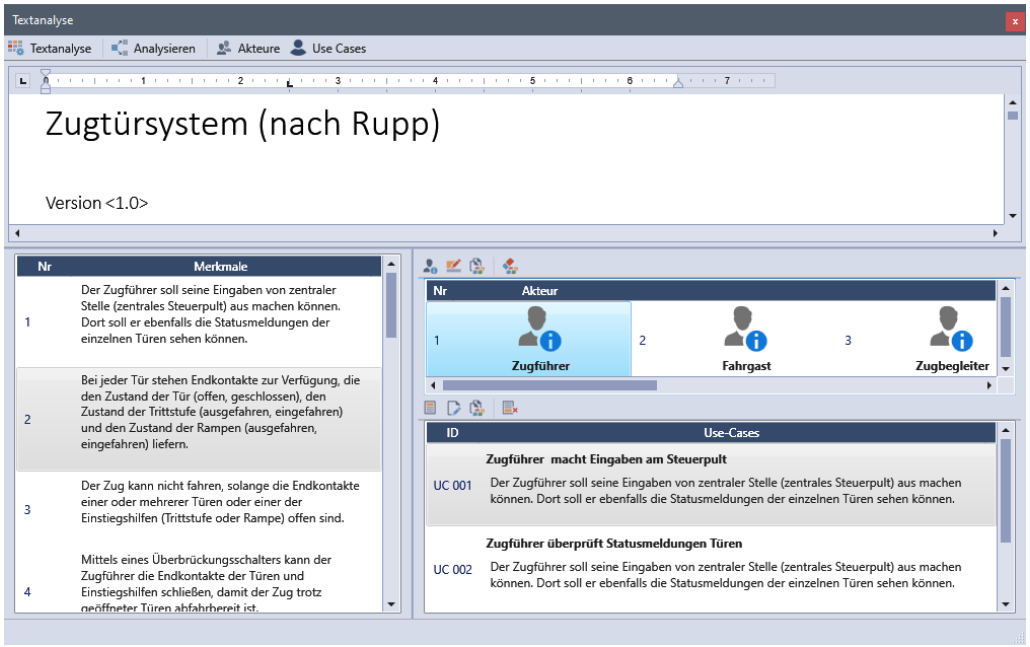


Abbildung 6.30: Dialog nach Ausführung der Textanalyse. Ausgewählte bzw. erstellte Use-Cases und Akteure können direkt in die Navigationsstruktur übernommen werden.

6.9.2 *Vorschläge für einen strukturierten Ablauf eines Use-Cases*

Bezogen auf die von den identifizierten Schwierigkeiten zur Analyse von Use-Case-Beschreibungen können demnach folgende High- Level-Anforderungen an eine Modellierungsumgebung definiert werden:

- Die Software sollte eine Unterstützungsmöglichkeit zur Analyse des Ablaufs einer Use-Case Beschreibung bieten.
- Die Software sollte die Möglichkeit bieten, relevante Textausschnitte für Use-Cases anzuzeigen.
- Die Software sollte die Möglichkeit bieten, Textausschnitte in die strukturierte Use-Case Beschreibung zu übernehmen.

Designentscheidungen:

- Um dieses Hilfsmittel nutzen zu können, muss eine Use-Case-Beschreibung in natürlicher Sprache vorliegen.
- Die Funktionalität zur Analyse des Textes wird über die Einbindung des OpenNLP (wie bereits einleitend beschrieben) realisiert, mit Hilfe dessen natürlichsprachliche Texte in einzelne Sätze zerlegt werden können. Sätze, die einen Konjunktiv enthalten und nicht Präsens formuliert sind, werden rot markiert. Der Nutzer wird somit darauf hingewiesen, dass diese nicht den Ablauf eines Use-Cases beschreiben {Abb. 6.32}.
- Es werden keine Sätze entfernt, da möglicherweise Nutzer trotzdem einen der scheinbar unbedeutenden Sätze nutzen möchten.
- Vorschläge können vom Nutzer in das Template übernommen werden.

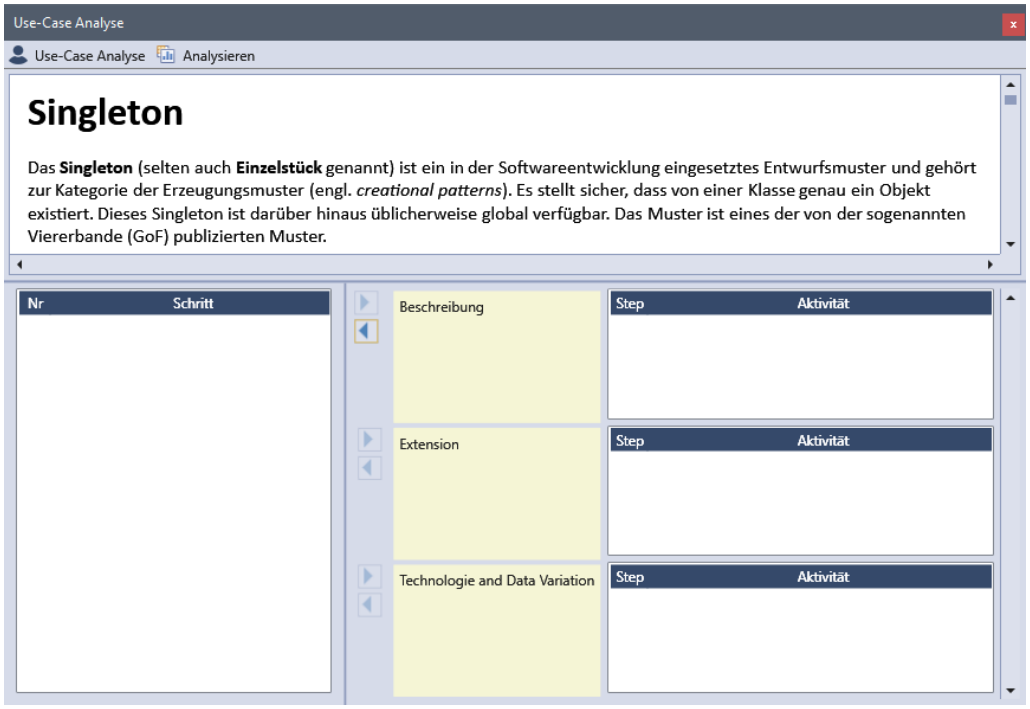


Abbildung 6.31: Dialog zum Scaffold Use-Case-Analyse, in der Phase, in der noch keine Analyse durchgeführt wurde

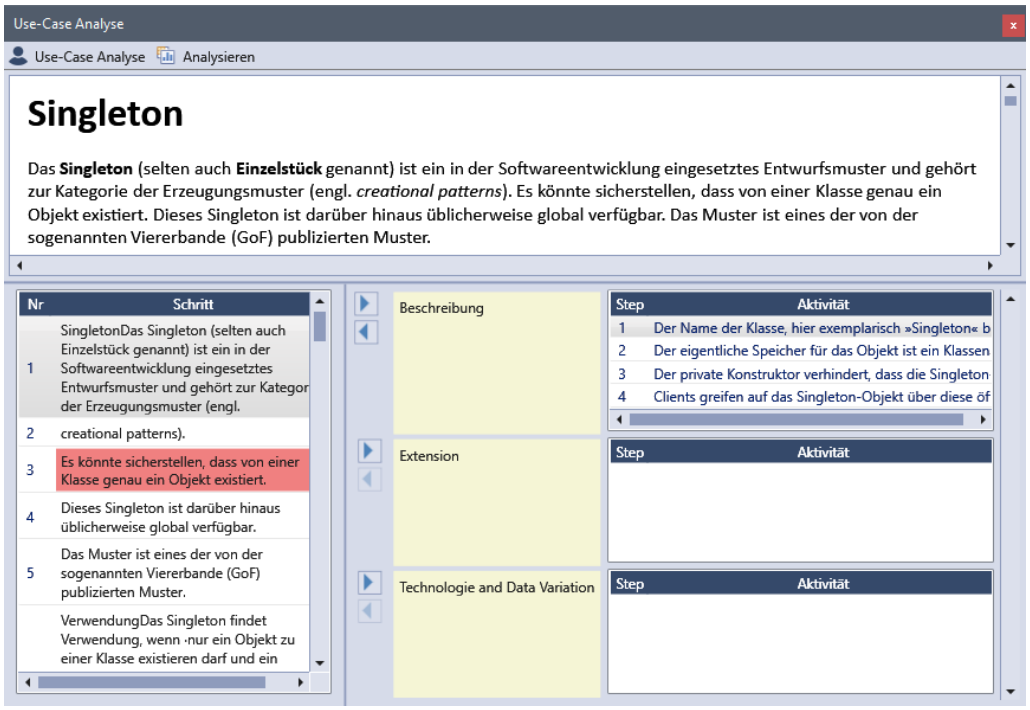


Abbildung 6.32: Screenshot nach Analyse

6.9.3 *Vorschläge zur Identifikation von Klassenkandidaten und Methoden*

Abschnitt 6.9.1 und Abschnitt 6.9.2 unterstützen den ersten Schritt des Makroprozesses Abschnitt 4.3. Die Abbott-Technik (Abschnitt 4.5) unterstützt den zweiten Schritt der „Ableitung von Klassen aus den Use-Cases“ (Balzert und Balzert (2009), 559; Abschnitt 4.3).

Um Studierende in dieser anzuleiten bzw. zur Nutzung dieser anzuregen, wurden folgende High-Level-Anforderungen abgeleitet:

- Die Software sollte eine Unterstützungsmöglichkeit zur Analyse von Substantiven und Verben einer strukturierten Use-Case-Beschreibung bieten.
- Die Software sollte die Möglichkeit bieten, Substantiv und Verben farblich zu markieren.
- Die Software sollte die Möglichkeit bieten, Substantive in ein Klassendiagramm zu übernehmen.

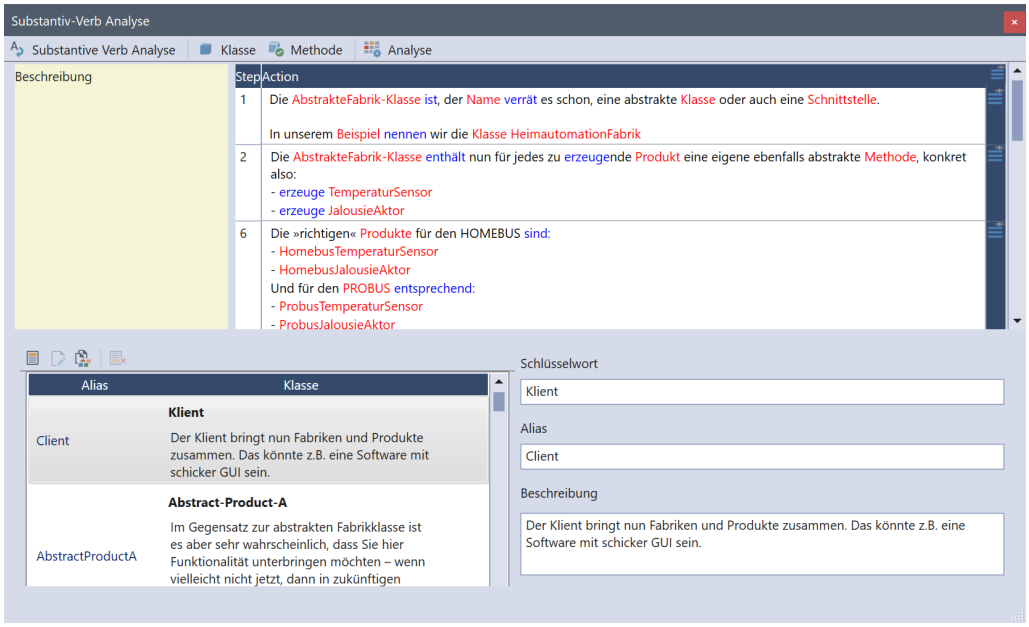


Abbildung 6.33: Dialog zum Scaffold Substantiv-Verb-Analyse

Designentscheidungen:

- Für die Substantiv-Verb-Analyse wird die Ablaufbeschreibung der strukturierten Use-Case Beschreibung herangezogen.
- Die Substantiv-Verb Analyse wird über die Einbindung des OpenNLP realisiert, mit Hilfe dessen Substantive und Verben in Sätzen identifiziert werden können.

- Substantive werden rot markiert, Verben werden blau markiert.
- Vorschläge für Klassen können per Drag-and-Drop in das Klassendiagramm übernommen werden und werden dort als Klassen dargestellt.

Abbildung 6.33 zeigt eine mögliche prototypische Realisierung der Anforderungen.

6.9.4 *Einsatzkonzept der Analysehilfen*

Die in Abschnitt 6.9.1,Abschnitt 6.9.2 und Abschnitt 6.9.3 vorgestellten Scaffolds können, sofern konsequent angewandt, bis zu einem ersten Grobentwurf für ein Klassendiagramm durchgeführt werden. Nachfolgend findet sich ein möglicher Verlauf, der zu einem Grobentwurf führt. Das hier gezeigte Beispiel ist angelehnt an den Vorschlag von Rupp et al. (2012). Es sei hier angemerkt, dass dies ein konstruiertes Beispiel ist, und ohne die manuelle Korrektur bzw. durch manuelles Hinzufügen von Use-Cases durch den Nutzer nicht die identische Lösung wie in Rupp et al. (2012) generiert wird. Wie beschrieben stellen die Analysen strukturierende Hilfsmittel für den Nutzer dar:

1) Zunächst wird die Textanalyse verwendet. Die Tabelle des Vision-Dokuments wird ausgelesen und Vorschläge für mögliche Use-Cases und Akteure gemacht (siehe Abb. 6.34).

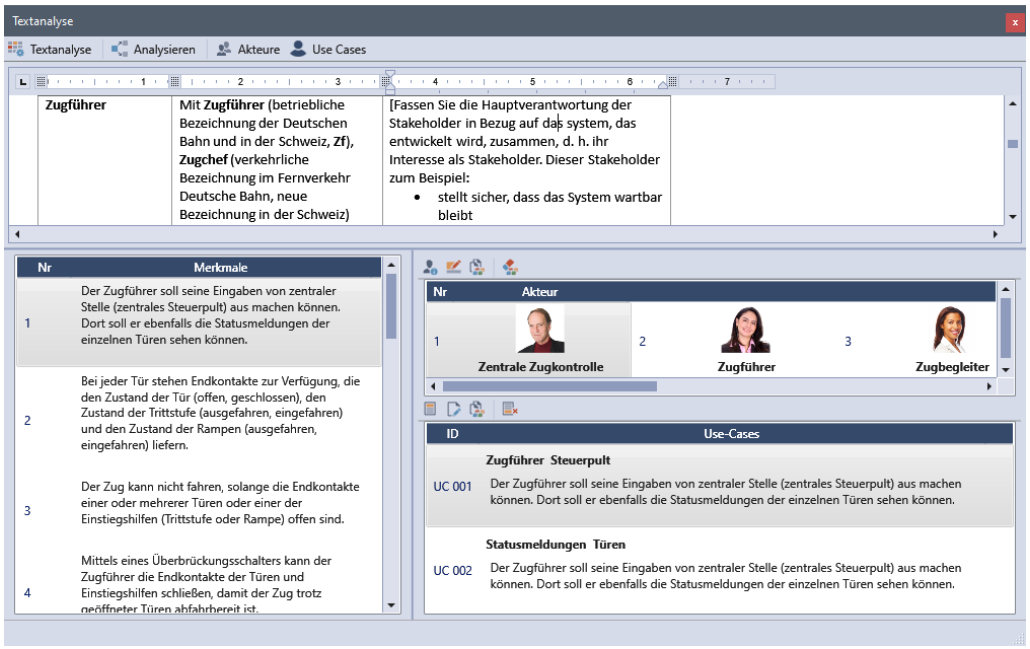


Abbildung 6.34: Identifikation relevanter Textstellen für das Zugtürsystem

2) Nach Bearbeitung der Vorschläge für die Use-Case-Beschreibungen und der Auswahl der Akteure können Use-Cases in das Use-Case-Diagramm per Drag-and-Drop übernommen werden Abb. 6.35.

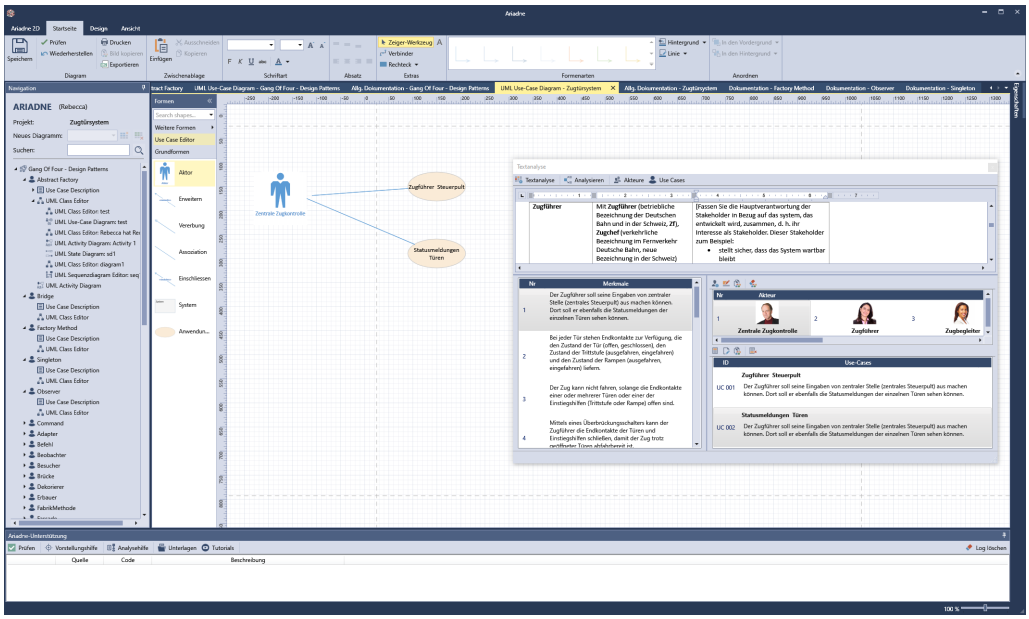


Abbildung 6.35: Erstellen von Use-Cases mit Hilfe der Analysehilfe

3) Mittels einer weiteren Analyse können auf Basis von vorhandenen Kurzbeschreibungen für Use-Cases, strukturierte Use-Case-Beschreibungen erstellt werden (Abb. 6.36).

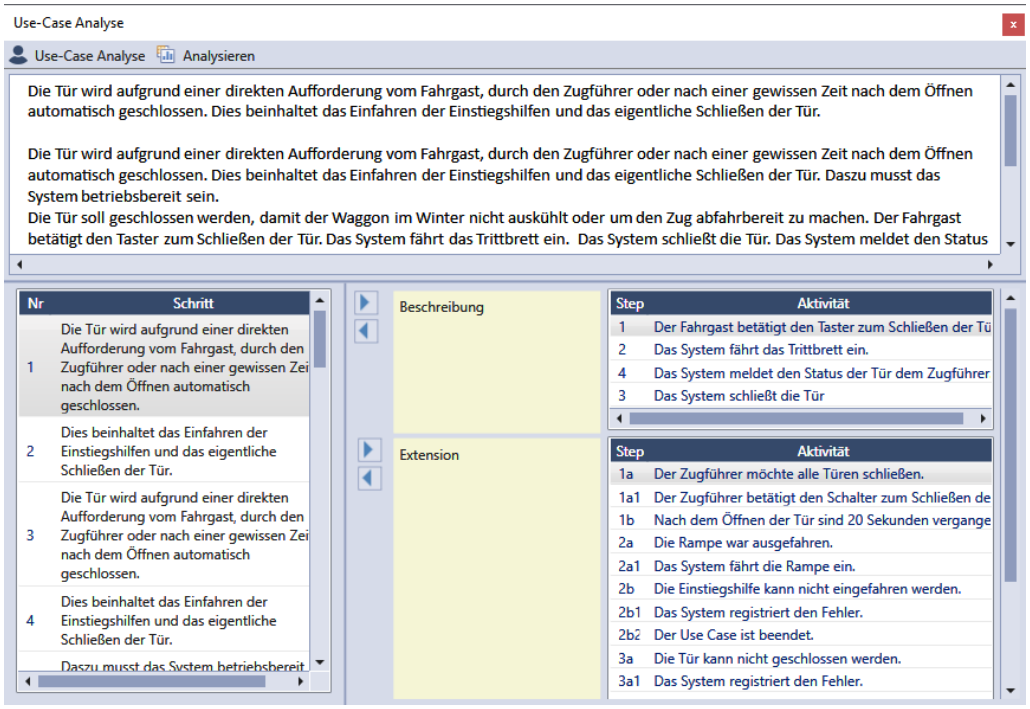


Abbildung 6.36: Analysehilfe für den Use-Case „Türe schließen“

Ein mögliches Ergebnis für eine strukturierte Use-Case Beschreibung findet sich in Abb. 6.37.

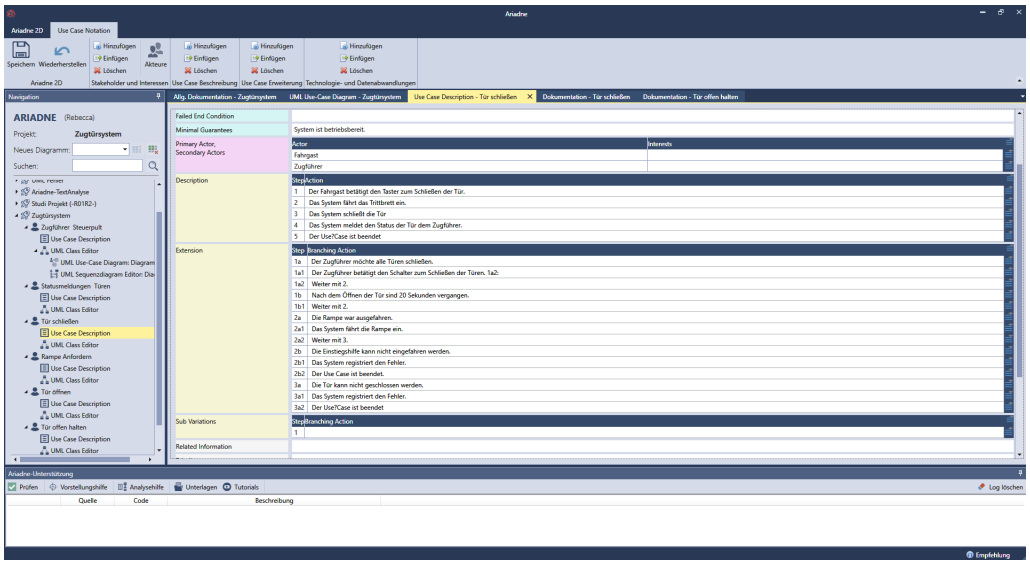


Abbildung 6.37: Use-Case-Template für den Use-Case „Türe schließen“

4) Auf Basis des strukturierten Use-Cases „Tür schließen“ kann nun eine Substantiv-Verb-Analyse durchgeführt werden (Abb. 6.38).

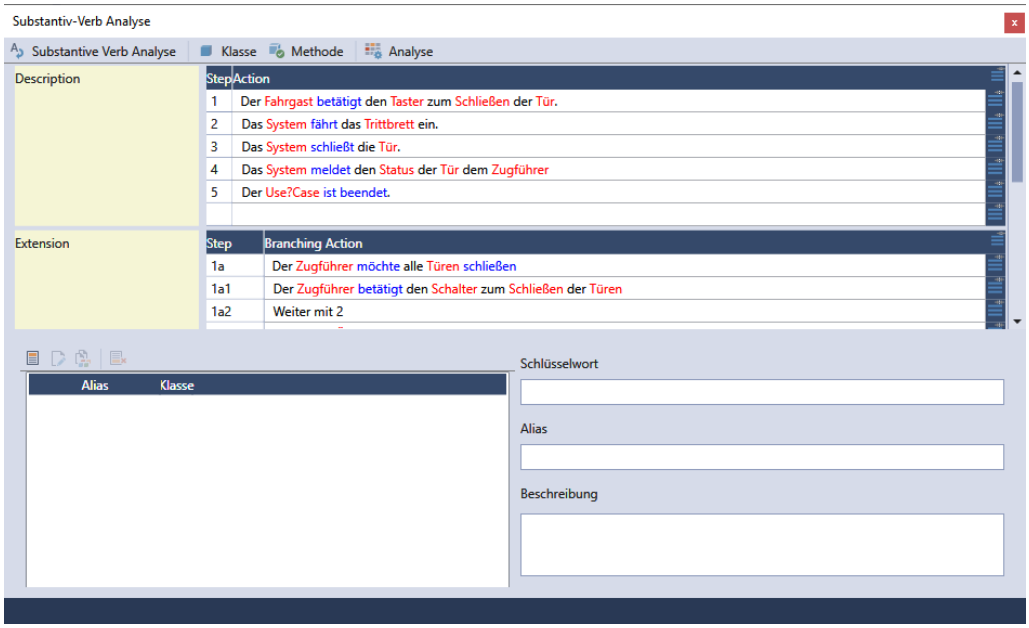


Abbildung 6.38: Erstellen von Klassenkandidaten mittels der Analysehilfe

Aus diesem Dialog heraus, können wie auch bereits bei den Vorschlägen zu Use-Cases, Klassen in das Klassendiagramm per Drag-and-Drop übernommen werden.

6.9.5 *Vorschläge als Sammlung existierender Projekte*

Thomasson et al. (2006a), Rosson und Carroll (1996) und Linn (1992) beschreiben in ihren Arbeiten die Relevanz der Sammlung von bestehenden „Lösungen“ bzw. Vorgehensweisen. Linn (1992) sieht in einer **Bibliothek an existierenden Beispielen und Templates** die Unterstützungsmöglichkeit für Studierende, da diese dann weniger Zeit damit verbringen, nach Beispielen zu suchen. Damit ist dieses Scaffold, wie auch schon die integrierten Unterlagen als Maßnahme zu verstehen, die ressourcenbezogenen Hindernisse adressiert. Die Idee einer Sammlung mit bereits existierenden Projekten wurde aufgegriffen und deshalb folgende Anforderungen und Designentscheidungen getroffen.

Anforderungen:

- Die Software sollte die Möglichkeit bieten, existierende Projekte als Sammlung von Beispielen bereitzustellen.
- Die Software sollte die Möglichkeit bieten, Diagramme und Text aus bereitgestellten Projekten zu kopieren.

Designentscheidungen:

- Die Software bietet Dozierenden die Möglichkeit, Projekte für alle Nutzer bereitzustellen.
- Nutzer, die die Rechte eines Studierenden haben, können die Projekte nicht bearbeiten aber Inhalte kopieren.
- Eine Suchfunktion innerhalb der Projekte ermöglicht die Suche nach Use-Cases über alle zur Verfügung stehenden Projekte.

6.10 AUGMENTED REALITY

Wie bereits im einleitenden Abschnitt 6.1.2.1 zu Erweiterter Realität (engl.: Augmented Reality, kurz: AR) angedeutet, wurden neben den gewünschten Scaffolds der Studierenden auch drei statische technologiebasierte Scaffolds im Rahmen von Kontrollgruppenexperimenten evaluiert, in denen Potenziale für mögliche neue Hilfestellungen vermutet wurden. Dazu zählen zwei Experimente zum Einsatz von AR und eines zum Einsatz von eines Eye-Movement Modeling Examples (Abschnitt 6.11). Alle drei Experimente wurden gemeinsam mit Masteranden durchgeführt. Die folgenden Ausführungen sind bereits in Reuter, Hauser, Muckelbauer et al. (2019)[30] und Reuter, Knietzsch, Hauser und Mottok (2019) publiziert und werden hier in ähnlicher Weise dargestellt.

6.10.1 *AR als neue Interaktionsmöglichkeit zur Steigerung von Motivation und Lernerfolg*

Das erste Experiment wurde durchgeführt, um den Einsatz von Augmented Reality zu evaluieren und damit eine neue Möglichkeit der Interaktion zu bieten. Auch die Cognitive Load Theory Abschnitt 3.2.1 und mögliche Überlastungen der Studierenden in der Nutzung zu komplexer existierender Software und daraus resultierende didaktische Lernhindernisse (Abschnitt 3.2) spielten eine Rolle. Dazu wurde für die AR-Brille Microsoft Hololens (2017) eine Benutzerschnittstelle zur Modellierung von Klassendiagrammen geschaffen, die im Rahmen des Experiments evaluiert wurde.

Evaluiert wurden neben der Software die Metriken Leistung und Motivation der Studierenden. Die Studie wurde als Kontrollgruppenexperiment konzipiert: Die Versuchsgruppe bearbeitete Aufgaben mit Hilfe des AR-Modellierungstools, die Kontrollgruppe verwendete eine funktionsgleiche PC-Software (Reuter, Hauser, Muckelbauer et al., 2019).

30 CC BY-NC-ND 4.0

6.10.1.1 *Modellierung von UML-Klassendiagrammen mit Augmented Reality vs. Modellierung von UML-Klassendiagrammen in Ariadne*

Ariadne, deren Name stellvertretend steht für den Ariadne-Faden durch den Softwareentwicklungsprozess, ist eine selbst entwickelte Software, die nach erfolgreicher Evaluation, sowohl 2D- als auch AR-UML-Modellierung integrieren soll.

In der Entwurfsphase können die Studierenden sowohl den 2D-UML-Editor als auch den AR-UML-Editor verwenden, die beide mit demselben Werkzeugpaket ausgestattet sind. Die Idee besteht darin, alle vorhandenen und relevanten Informationen im Zusammenhang mit den Modellierungselementen eines Systems wiederzuverwenden. Diese wichtigen Daten werden mit den entsprechenden Modellierungselementen verknüpft und der Benutzer sollte in der Lage sein, sie in der dritten Dimension zu erforschen und mit ihnen zu interagieren (Reuter, Hauser, Muckelbauer et al., 2019).

SYSTEM-ÜBERSICHT Ariadne besteht für das Experiment aus zwei Modulen: Die 2D-Umgebung und die AR-Umgebung. Die Desktop- und die AR-Softwareversion teilen sich eine Schnittstelle für den Zugriff auf Daten und Ergebnisse in einer gemeinsamen Datenbank. So können die Studierenden sowohl mit der Desktop-Version als auch mit der AR-Version der Software arbeiten und nahtlos zwischen beiden Versionen wechseln. Die 2D-Umgebung ist eine WPF-Anwendung (Windows Presentation Foundation). Die AR-Umgebung wird mit der Unity Spielengine für die AR-Brille Microsoft Hololens implementiert (Reuter, Hauser, Muckelbauer et al., 2019).

Mit Ariadne sollen die Studierenden mit der notwendigen Komplexität zur Modellierung eines Softwaresystems konfrontiert werden. Dazu stellt Ariadne einen minimalen (unbedingt notwendigen) Umfang an Funktionalität zur Verfügung. Diese Reduktion der Funktionalität erlaubt es den Studierenden, sich explizit auf ihre Aufgabe zu konzentrieren (Reuter, Hauser, Muckelbauer et al., 2019).

INTERAKTIONSGESTALTUNG Dieser Abschnitt behandelt das Interaktionsdesign der AR-UML-Modellierungssoftware. Während die 2D-Benutzeroberfläche bekannten UML-Editoren ähnelt, erstrecken sich die Bedienkonzepte für Desktop-Computer nicht trivial auf den Bereich der AR-Benutzeroberflächen.

Wie bereits in Abschnitt 5.3.5 und Abschnitt 6.2 beschrieben, ergibt sich ein Hauptproblem beim Modellieren aus der Komplexität der Erstellung und Bearbeitung der Modelle in gängiger Software. Deshalb

wurde der Ansatz gewählt, die Benutzeroberfläche so weit wie möglich zu vereinfachen, indem die Werkzeuge so ausdrucksstark wie möglich gehalten wurden. Dies wurde auch als ein Problem beim Erlernen der UML erwähnt (Reuter, Hauser, Muckelbauer et al., 2019; Siau & Loo, 2006).

Bei der Verwendung der AR-Brille und der Nutzung von AR gibt es keine Maus, die man zum Zeigen oder zum Ausführen verschiedener Aktionen (Links- und Rechtsklick, Doppelklick usw.) verwenden kann. Vielmehr werden diese Aktionen im AR-Editor als Gesten realisiert, die einzigen verbleibenden Oberflächen-Schaltflächen sind die Speichern und Zurücksetzen-Schaltflächen, da diese nicht kontextsensitiv sind: Man kann nicht eine einzelne Klasse oder Beziehung speichern, sondern nur das Diagramm als Ganzes. Ein Beispiel für die Gesten im System ist in Abb. 6.39 dargestellt. Der Studierende sieht einen vergrößerten Teil der Realität beispielsweise durch die AR-Brille[31] (d. h. er sieht den Raum und die UML-Klassen im Raum schweben). Mit Hilfe der Gesten können Elemente angelegt werden und Eigenschaften des Diagramms geändert werden (Reuter, Hauser, Muckelbauer et al., 2019).

Für den AR-Editor wurde eine schlanke Oberfläche angeboten, die nur eine Schaltfläche zum Zurücksetzen und Speichern enthält. Dies geschah, um so viel Modellierungsfläche wie möglich zur Verfügung zu stellen und Komplexität durch zusätzliche Schaltflächen zu reduzieren. Die restliche Funktionalität wurde während der Modellierung über Gesten und Dropdown-Menüs realisiert (Reuter, Hauser, Muckelbauer et al., 2019).

Der Benutzer sieht zunächst nur einen leeren Bereich, eine Schaltfläche zum Zurücksetzen und Speichern. Die Modellierung erfolgt über Gesten. Für neue Benutzer werden im Vorfeld ein Tutorial und ein Spickzettel zur Verfügung gestellt, damit sie sich mit den Bedienelementen vertraut machen können. Im Gegensatz zur virtuellen Realität sieht der Benutzer seine reale Umgebung, weshalb er bei der Modellierung auch den Spickzettel verwenden kann (Reuter, Hauser, Muckelbauer et al., 2019).

PROTOTYP-IMPLEMENTIERUNG Der Prototyp des 2D-Editors funktioniert wie man es von einem UML-Klassendiagramm-Editor erwarten würde: Es gibt eine Toolbox mit den gegebenen Elementen (Klassen, Schnittstellen, Vererbung und Assoziationsbeziehungen). Der Benutzer kann neue Elemente durch einen Klick auf ein gegebenes Elementsymbol und durch einen zweiten Klick auf die gewünschte Position im Modellierungsbereich erstellen (Reuter, Hauser, Muckelbauer et al., 2019).

31 Für das Experiment stand die Microsoft Hololens zur Verfügung, es existieren weitere AR-Brillen auf dem Markt.

Um eine Verbindung zwischen zwei Klassen herzustellen, muss der Benutzer zunächst den gewünschten Verbindungstyp auswählen. Dann muss er/sie auf die Quellklasse und als nächstes auf die Zielklasse klicken (Reuter, Hauser, Muckelbauer et al., 2019).

Diagramme können durch einen Klick auf die Schaltfläche „Speichern“ in der Datenbank gespeichert werden. Durch Doppelklick auf Klassen oder Interfaces können neue Attribute und Methoden hinzugefügt werden. Diese Funktion ist auch durch einen Rechtsklick auf diese Elemente verfügbar (Reuter, Hauser, Muckelbauer et al., 2019).

Abbildung 6.39: Nutzerdemonstration während der Modellierung eines Klassendiagramms mit Gesten

Im AR-Editor werden Grundfunktionalitäten, wie das Erstellen einer neuen Klasse, durch verschiedene Gesten realisiert. Wo mehrere Optionen möglich sind (z. B. die Einstellung, ob eine Beziehung zwischen zwei Klassen eine Spezialisierung oder eine Assoziation sein soll), werden die möglichen Optionen in einer Dropdown-Menü präsentiert. Der Benutzer kann dann aus diesem Menü die gewünschte Option auswählen. Diese Aktion wurde über ein Dropdown-Menü zu realisiert, da dies die Anzahl der Gesten, die ein Benutzer ausführen muss, um die gewünschte Aktion auszuführen, deutlich reduziert. In Abb. 6.40 ist ein Dropdown-Menü in Aktion dargestellt. Der Benutzer kann die Sichtbarkeit bestimmter Merkmale einer abstrakten Klasse festlegen (Reuter, Hauser, Muckelbauer et al., 2019).

Abbildung 6.40: Beispiel für ein Dropdown-Menü zur Zuweisung der Sichtbarkeit von Attributen in einer abstrakten Klasse[32]

Der folgende Abschnitt beschreibt die in der Studie verwendete Methodik.

6.10.1.2 *Methodik*

Da die Software bereits für zwei Varianten, PC (2D) und Hololens (Augmented Reality) ausgelegt ist, wurde ein kontrollgruppen-basiertes Experiment konzipiert, das, wie beabsichtigt, eine differenzierte Analyse der beiden getesteten Softwarelösungen ermöglicht und damit eine Evaluation von AR als neue Interaktion in der Modellierung zulässt (Reuter, Hauser, Muckelbauer et al., 2019).

BESCHREIBUNG DER ZIELGRUPPE Die Zielgruppe waren Teilnehmer eines Software Engineering-Kurses im Wintersemester 2017/2018, der als Wahlpflichtfach für Ingenieure angeboten wird. Die Studenten waren Novizen in Bezug auf die UML-Modellierung. Sie hörten vor diesem Kurs über Klassendiagramm-Modellierung nur eine Lehreinheit. Diese Vorlesungseinheit hat im Rahmen einer Lehrveranstaltung über objektorientierte Programmierung im zweiten Semester stattgefunden (Reuter, Hauser, Muckelbauer et al., 2019).

BESCHREIBUNG DER LEHR-/LERNSETTINGS Das Modul besteht aus einer Vorlesung und einer vorlesungsbegleitenden Übung. Das Experiment war vollständig in den Lehrplan integriert, d. h. die Studierenden

32 Diese Abbildung stammt aus der Unity-Umgebung und nicht aus dem realen System in Aktion. Daher ist der Hintergrund in einem bläulichen Farbverlauf dargestellt.

hörten die Vorlesung zu UML-Modellierung und belegten eine Woche später die zugehörige praktische Präsenz-Übung, in der das Experiment stattfand (Reuter, Hauser, Muckelbauer et al., 2019).

ziel Ziel dieses Experiments war es, herauszufinden, ob die Versuchsgruppe des Experiments...

- erfolgreicher, ähnlich erfolgreich,
- motivierter, ähnlich motiviert,
- inhaltlich interessierter, inhaltlich- ähnlich interessiert, ... ist, als die Kontrollgruppe (Reuter, Hauser, Muckelbauer et al., 2019).

verfahren Um einen Eindruck davon zu bekommen, wie sich die Teilnehmer nach der Vorlesung mit der UML-Klassenmodellierung einschätzen und um nach möglichen Vorkenntnissen zu fragen, wurde direkt nach der Vorlesungseinheit- also eine Woche vor dem Experiment - eine Selbsteinschätzung mittels eines Fragebogens durchgeführt. Die Studierenden wurden über das bevorstehenden Experiment dabei nicht in Kenntnis gesetzt (Reuter, Hauser, Muckelbauer et al., 2019).

Für das Experiment wurden die Studierenden nach dem Zufallsprinzip den beiden Gruppen (experimentell (AR), Kontrolle (PC-2D)) zugeteilt. Die Gruppen wurden in drei verschiedene Räume aufgeteilt: Die 2D-Gruppe löste die Aufgabe in Einzelarbeit in einem Computerraum. Die AR-Gruppe wurde in die übrigen zwei Räume aufgeteilt, da für das Experiment nur zwei AR-Brillen zur Verfügung standen und wir die wartenden Teilnehmer nicht beeinflussen wollten. Sie erhielten eine nicht themenbezogene Aufgabe, an der sie während der Wartezeit arbeiten konnten (Reuter, Hauser, Muckelbauer et al., 2019).

Beide Gruppen mussten das Tutorial für die Software (Ariadne 2D oder Ariadne AR) durcharbeiten, bevor sie sich mit der Aufgabe befassen konnten. Danach erhielten die Teilnehmer die identische Aufgabe, ein Computerschachprogramm für beide Gruppen zu modellieren. Es wurde keine zeitliche Begrenzung festgelegt. Auftretende Softwareprobleme wurden protokolliert (Reuter, Hauser, Muckelbauer et al., 2019).

Nach Abschluss der Aufgabe erhielten alle Teilnehmer den Fragebogen MUSIC®-Inventory[33] (Jones, 2009), um ihre Motivation in Bezug auf das Experiment festzuhalten (Reuter, Hauser, Muckelbauer et al., 2019).

Die Teilnahme war freiwillig und von der regulären Bewertungsübersicht ausgeschlossen. Darüber hinaus wurden alle Daten (Fragebogen zur Selbsteinschätzung, Einreichung der Aufgabe, MUSIC®-Inventory)

33 Erlaubnis zur Verwendung des Fragebogens für das Experiment vorhanden.

in anonymisierter Form eingereicht. Zusammenfassend wurden folgende Erhebungen durchgeführt (Reuter, Hauser, Muckelbauer et al., 2019):

1. Fragebogen zur Selbsteinschätzung (eine Woche bevor das Experiment stattfand)
2. Aufgabe der Arbeitsgruppe
3. Interview (direkt nach Abschluss der Aufgabe)
4. MUSIC®-Inventory

FRAGEBOGEN: ZUR SELBSTEINSCHÄTZUNG Der Fragebogen zur Selbsteinschätzung wurde selbst erstellt und bestand aus 15 Items. Die Items wurden mit Bezug auf die Lernziele der Vorlesungs- und Übungseinheit auf der Basis von Modellierung von Klassendiagrammen erstellt. Grundlage dafür war die Lernzieltaxonomie SOLO (Structure of the Observed Learning Outcome) von Biggs, Collis und Edward (1982) bzw. die adaptierte Version von Brabrand und Dahl (2009), mit deren Hilfe es möglich ist, ergebnisorientierte Lernziele zu formulieren. Die SOLO-Taxonomie hat fünf Stufen von Inkompetenz (SOLO 1) bis zur Fähigkeit, Wissen auf ein neues Gebiet zu verallgemeinern (SOLO 5). Sie wird verwendet, um die Lernergebnisse der Studenten im Hinblick auf ihre Komplexität zu qualifizieren. Wir haben bewusst nur niedrige SOLO-Niveaus verlangt, da es sich um Neulinge auf diesem Gebiet handelt (Reuter, Hauser, Muckelbauer et al., 2019). Folgend finden sich einige Beispiele von Lernzielen:

Studierende können...

- ...den Pfeiltyp einer Komposition identifizieren (SOLO 2).
- ...verschiedene Arten von Vereinigungen unterscheiden (SOLO 4).
- ...eine Aggregationsverbindung korrekt verwenden (SOLO 3) (Reuter, Hauser, Muckelbauer et al., 2019).

FRAGEBOGEN: MUSIC® INVENTORY Das MUSIC® Model of Academic Motivation Inventory (MUSIC® Inventory) ist ein Fragebogen zur Beurteilung der Wahrnehmung der MUSIC®-Komponenten (Empowerment, Usefulness, Success, Interest and Caring) durch die Studierenden für eine Aktivität in einem Kurs oder einen gesamten Kurs (Jones, 2009). Der MUSIC®- Inventory ist ein forschungsbasierter Fragebogen, der zuverlässige und valide Ergebnisse mit Skalen liefert, die durch eine konfirmatorische Faktorenanalyse nachgewiesen wurden. Bisher wurden diese Ergebnisse in englisch-, isländisch-, arabisch- und spanischsprachigen Ländern mit Studierenden der Elektrotechnik, Systemtechnik und des Wirtschaftsingenieurwesens erzielt (Hussein, Bakar & Jones, 2015; Jones, 2009; Jones, Li & Cruz, 2017; Jones & Skaggs, 2016). In der vorliegenden

Arbeit wurde mit Erlaubnis des Autors eine deutsche Übersetzung des von Jones (2009) zur Verfügung gestellten MUSIC® Inventory verwendet (Reuter, Hauser, Muckelbauer et al., 2019), da dieser durch seine bereits nachgewiesene Reliabilität und Validität reliable und valide Ergebnisse liefert.

Der Fragebogen besteht aus fünf Komponenten, die bei der Gestaltung einer Instruktion zu berücksichtigen sind, nämlich „Empowerment, Usefulness, Success, Interest and Caring". Jede Komponente wurde aus pädagogischer und psychologischer Forschung und Theorie abgeleitet. Anhand des Beispiels Autonomie bedeutet dies, Autonomie ist ein verwandtes Konstrukt von Empowerment, weshalb Empowerment den Grad beschreibt, in dem ein/e Studierende/r wahrnimmt, dass er/sie die Kontrolle über sein/ihr Lernumfeld im Kurs hat (Reuter, Hauser, Muckelbauer et al., 2019).

Dasselbe gilt für - die Nützlichkeit der Kursarbeit für die Zukunft der Studierenden (Nützlichkeit), - die Wahrnehmung, in der Kursarbeit erfolgreich zu sein (Erfolg), - das Interesse an den Lehrmethoden und der Kursarbeit (Interesse), - das Empfinden eines Studierenden gegenüber seines Dozierenden (Kümmert der Dozierende sich um den Erfolg in der Kursarbeit und um das Wohlbefinden des Studierenden (Fürsorge)) (Jones, 2009; Reuter, Hauser, Muckelbauer et al., 2019). Wenn ein Dozent eine oder mehrere dieser Komponenten unterstützt, erhöht sich die Motivation der Studierenden und damit auch der Lernerfolg der Studierenden. Die Items des Fragebogens wurden auf einer Sechs-Punkte-Likert-Skala bewertet, die von stark abweichend bis stark zustimmend reichte (Reuter, Hauser, Muckelbauer et al., 2019).

PERFORMANCE (LEISTUNG) DER STUDIERENDEN Die Leistung der Studierenden wurde anhand der vorgegebenen Aufgabe gemessen. Die Studierenden konnten bei der Aufgabe bis zu 100 Punkte erreichen. Für die Punktevergabe wurden Punkte auf der Grundlage der gültig modellierten Klassen, der verwendeten Beziehungen, der modellierten Attribute und Methoden vergeben (Reuter, Hauser, Muckelbauer et al., 2019).

Tabelle 6.2: Vergebene Punkte für modellierte Elemente

Element	Punkte
Klasse	5
Assoziation	2
Methode	2

Element	Punkte
Attribut	2
Multiplizität	1

INTERVIEW Nach Abschluss der Aufgabe oder des Experiments führten wir mit jedem Teilnehmer der Versuchsgruppe ein leitlinienbasiertes Interview durch. Die Benutzerfreundlichkeit im Umgang mit der AR-Brille und die weitere Selbsteinschätzung über die Leistung in der Aufgabe standen im Mittelpunkt der Interviews. Wir führten halbstandardisierte Interviews mit drei offenen Leitfragen durch:

- Wie sind Sie mit dem Modellieren mit AR-Brille zurechtgekommen?
- Sollte es weitere Toolboxen für dieses Werkzeug geben? Beschreiben Sie Ihre Erfahrungen mit der Modellierung/ Verwendung mit der AR-Brille
- Wie viele Punkte würden Sie sich für diese Aufgabe geben? In Form von 100 Punkten: Wie würden Sie sich selbst bewerten? (Reuter, Hauser, Muckelbauer et al., 2019)

6.10.1.3 *Auswertung*

Insgesamt (*N=14*) Studierende aus dem 5. Semester nahmen an dem Experiment teil. Trotz dieser sehr kleinen Stichprobengröße sind dies 100% der Studierenden, die auch die Vorlesung besuchen (d. h. die Grundgesamtheit für diese Studie, es hat keine Stichprobenziehung stattgefunden). Dementsprechend gab es jeweils sieben Teilnehmer für die Experimentalgruppe und die Kontrollgruppe und vier Artefakte (beschrieben in den Unterabschnitten des Methodenteils) pro Teilnehmer, die ausgewertet werden können (Reuter, Hauser, Muckelbauer et al., 2019).

Für die Auswertung wurden die Daten zweimal gruppiert: Erstens bezüglich der 2D- oder AR-Zuordnung und zweitens bezüglich des Mittelwertes in der Selbsteinschätzung: Es konnten zwei Gruppen gebildet werden, eine mit einem Mittelwert von *2,73* und niedriger, eine mit einem Mittelwert von *2,87* und höher (die Studierenden bewerteten auf einer Sechs-Punkte-Likert-Skala, die von stark ablehnend bis stark zustimmend reichte) (Reuter, Hauser, Muckelbauer et al., 2019).

Ein signifikanter Unterschied wurde für den Punkt *Interest* an der MUSIC®-Skala gefunden: Ein T-Test ergab, dass die Gruppe mit geringeren Vorkenntnissen einen Mittelwert von *3,86 (SD = 0,18)* aufwies, die Gruppe mit mehr Vorkenntnissen einen Mittelwert von *4,43 (SD = 0,56)*, $t(12) = -2{,}57$, $p = .024$.

Tabelle 6.3: Zuverlässigkeit für jede Skala (Cronbach's Alpha), Mittelwerte und Standardabweichungen berechnet für Experimentalgruppe und Kontrollgruppe für das MUSIC® Modell

MUSIC® Skala	M 2D Gruppe (SD) N= 7	M AR Gruppe (SD) N= 7	Cronbach's α^a N= 14 ^{a} Cronbach's α = .835 ohne Item 19
Empowerment	3.89 (.45)	3.94 (.76)	.821
Usefulness	4.40 (.50)	4.51 (.45)	.337 *
Success	4.25 (.92)	4.61 (.57)	.841
Interest	4.07 (.29)	4.21 (.66)	.834
Caring	4.55 (.23)	4.64 (.56)	.749

Die AR-Gruppe erzielte in den MUSIC® -Skalen höhere Werte als die 2D-Gruppe, aber aufgrund der kleinen Stichprobe von sieben Teilnehmern pro Gruppe wurden keine signifikanten Unterschiede oder Korrelationen festgestellt. Dies könnten jedoch Indikatoren für eine höhere Motivation bei der Verwendung von AR sein (siehe Tabelle 6.3) (Reuter, Hauser, Muckelbauer et al., 2019). Bei der Aufgabe, bei der 100 Punkte hätten erreicht werden können, erreichte die 2D-Gruppe einen Mittelwert von *79,43* (*SD = 9,74*) Punkte, während die AR-Gruppe *68,21* Punkte (*SD = 13,49*) erreichte. Jedoch war der Unterschied nicht signifikant *t(12) = 1,783*, *p = 0,1*. Ein möglicher Einflussfaktor könnte das Vorwissen der Kontrollgruppe gewesen sein. Im Fragebogen zur Selbsteinschätzung lag der Mittelwert der 2D-Gruppe bei *3,3* (*SD = 0,89*), der Mittelwert der AR-Gruppe bei *2,78* (*SD = 0,64*), was deuten darauf hin, dass die 2D-Gruppe mehr Vorkenntnisse hatte, aber dieser Unterschied ist nicht signifikant. In Abb. 6.41 sehen Sie ein Beispiel der Dozierendenperspektive in Ariadne 2D; die Abbildung zeigt das Modell, das von einem Studenten während der Studie konstruiert wurde. Abbildung 6.42 zeigt im Gegensatz dazu das Modell eines Studierenden in AR (Reuter, Hauser, Muckelbauer et al., 2019).

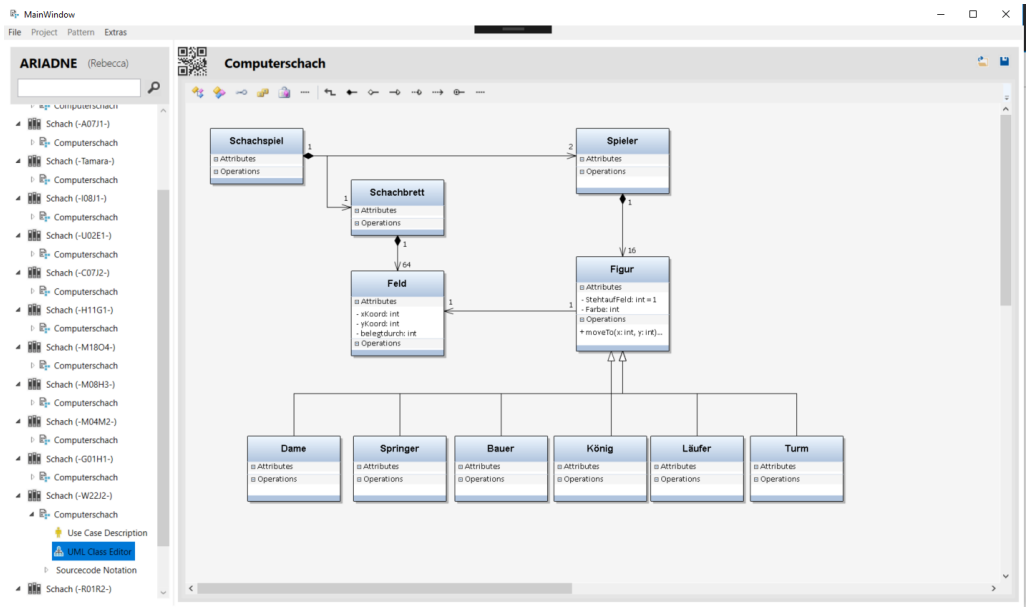


Abbildung 6.41: Die Perspektive des Dozierenden in Ariadne

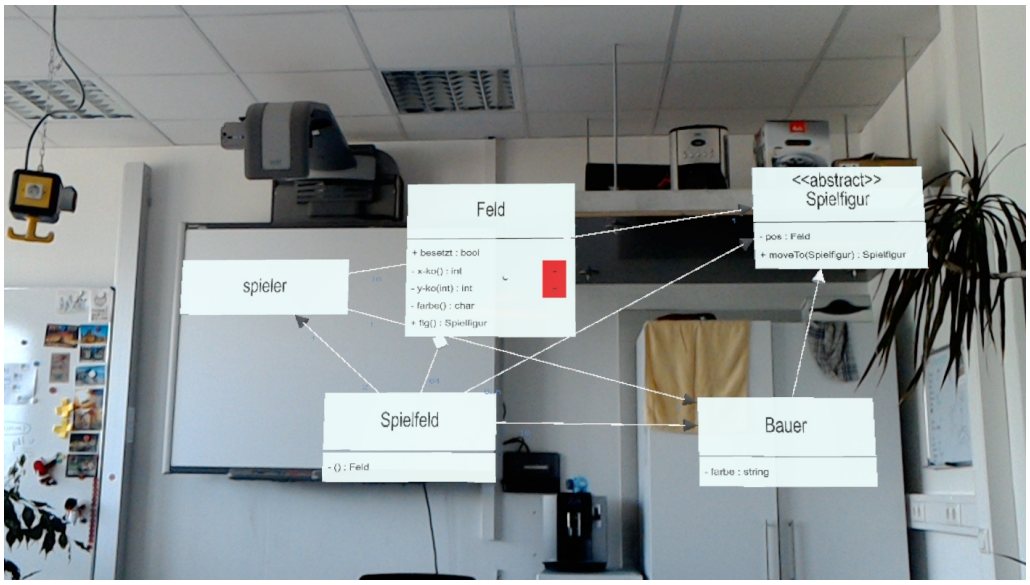


Abbildung 6.42: Studierendenabgabe während des Experiments

Um zu identifizieren, wie gut Studierende auf dem Gebiet der Modellierung abschneiden, wurde zusätzlich eine hierarchische Clusteranalyse durchgeführt. Die Clusteranalyse ist ein multivariates Verfahren, das zur Datenreduktion und zur Einteilung der Probanden in Gruppen oder Cluster verwendet wird. Sie ist der bekannteren Hauptkomponentenanalyse recht ähnlich, aber im Gegensatz zu diesem Verfahren ist sie stärker auf die Subjekte (und nicht auf die Items) fokussiert. Wenn eine Clusteranalyse durchgeführt wird, besteht ihr Ziel darin, Cluster zu bilden,

die sich intern sehr ähnlich sind, und sich aber gleichzeitig so weit wie möglich von den anderen Clustern unterscheiden. Beispiel: Intern haben alle erstellten Cluster oder Gruppen gemeinsame Attribute, die alle zugehörigen Subjekte mit den anderen Mitgliedern der Gruppen teilen. Wenn die Gruppen jedoch miteinander verglichen werden, unterscheiden sie sich deutlich von den anderen (Bortz & Schuster, 2010; Reuter, Hauser, Muckelbauer et al., 2019).

Als Variablen wurden die metrischen Variablen des Fragebogens zur Selbsteinschätzung im Mittelwert und die vergebenen Punkte verwendet. Die Clusteranalyse zeigt drei Cluster (siehe Abb. 6.43) bestehend aus 2 mal 5 und einmal 4 Fällen (für insgesamt 14 Teilnehmer).

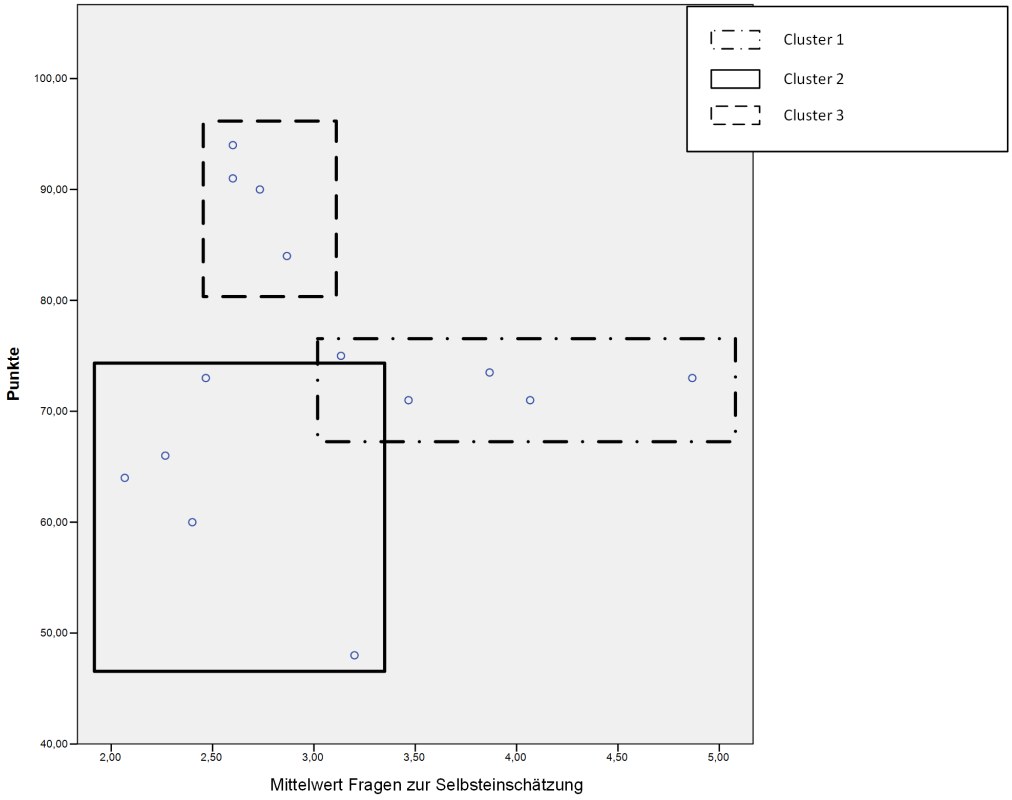


Abbildung 6.43: Ergebnisse der Clusteranalyse

Auffällig an Cluster 1 ist, dass der Mittelwert für die Selbsteinschätzung in den Antworten der Teilnehmer relativ hoch ist (> *3,13*). Außerdem erzielten sie *72,7* Punkte im Durchschnitt. Dieser Cluster setzt sich recht homogen aus Teilnehmern der 2D-Gruppe und Teilnehmern der AR-Gruppe zusammen. Cluster 2 erzielte im Durchschnitt *62,2* Punkte mit einer Bewertung von *2,4*. Dieser Cluster besteht hauptsächlich aus der Experimentalgruppe; nur ein Fall stammt aus der 2D-Gruppe (Reuter, Hauser, Muckelbauer et al., 2019).

Cluster 3 umfasst die Fälle mit den höchsten Punktzahlen. Dieser Cluster bewertete am schlechtesten (*2,6*). Hier besteht das Cluster hauptsächlich aus 2D-Fällen. Die Korrelation zwischen der Selbsteinschätzung und Punkten in der 2D-Gruppe bestätigt dies. Es gibt eine negative lineare Korrelation (*rbp= -.590*), die nicht signifikant ist (*p= .163*), aber sie könnte ein Indikator dafür sein, dass die Teilnehmer, die sich schlechter einschätzen, mehr Punkte erhielten als die anderen, die weniger Punkte erreichten (Reuter, Hauser, Muckelbauer et al., 2019).

Die Interviews wurden transkribiert und deduktiv codiert, d. h. abhängig der Fragestellung Kategorien abgeleitet: Nach allgemeiner Erfahrung mit AR, Benutzerfreundlichkeit und Selbsteinschätzung. In der Kategorie Erfahrung gibt es keine klare Tendenz; etwa die Hälfte der Teilnehmer gab an, dass die Handhabung schwierig sei, die andere Hälfte empfand die Handhabung als einfach. Viele Gründe konnten auf die ungewöhnliche Situation zurückgeführt werden. In der Kategorie Benutzerfreundlichkeit wurden hauptsächlich Probleme bei der Eingabe über die Tastatur genannt. Fast alle Testpersonen wünschten sich eine Toolbox/Toolbar. In der Kategorie Selbsteinschätzung schätzten sich die Teilnehmer eher unterdurchschnittlich ein (Reuter, Hauser, Muckelbauer et al., 2019).

6.10.1.4 *Einschränkungen & Methodische Grenzen*

Die Studie weist folgende Einschränkungen auf:

- Schlussfolgerungen basieren auf einer kleinen Stichprobe. Obwohl alle Studierenden des untersuchten Kurses an der Studie teilgenommen haben, ist $N = 14$ zu klein, um signifikante Ergebnisse zu erhalten und eine Gesamtschlussfolgerung zu ziehen.
- Die Studierenden waren Novizen in der Modellierung von Klassendiagrammen. Die Konfrontation mit einem neuen Werkzeug, einer neuen Technologie und einem neuen Lerninhalt kann zu Überforderung und kognitiver Überlastung führen.
- Zielgruppe waren Studierende der Nicht-Hauptfachrichtung Informatik. Bei Studierenden der Informatik können die Ergebnisse anders ausfallen.

6.10.1.5 *Diskussion*

Trotz der kleinen Stichprobe können wir eine Tendenz in der Motivation der Studierenden, die mit AR modelliert haben, im Vergleich zu denen, die mit 2D modelliert haben, feststellen. Es wurden keine signifikanten Korrelationen und Abweichungen zwischen Motivation und Leistung festgestellt werden. Motivation und Leistung ändern sich somit nicht

signifikant. Im Allgemeinen war jedoch die gesamte Gruppe sehr motiviert und der MUSIC®- Inventory zeigt deutlich überdurchschnittliche Mittelwerte. Darüber hinaus lässt sich kein signifikanter Unterschied in der Punktverteilung zwischen der experimentellen und der Kontrollgruppe feststellen. Das bedeutet, dass keine Gruppe signifikant besser (oder schlechter) abgeschnitten hat als die Andere.

6.10.2 *Augmented Reality und Abstraktionsfähigkeit*

Das zweite Experiment, das in den folgenden Abschnitten beschrieben wird, untersucht das von Akçayir et al. (2016) bereits festgestellte Potenzial von AR abstrakte Konzepte zu visualisieren für den Bereich des Requirements Engineering und der objektorientierte Analyse. Konkret war die Intention des Experiments, das Potenzial von AR zur Unterstützung des abstrakten Denkens bzw. die Fähigkeit des Abstrahierens zu überprüfen, die eine der wichtigsten Fähigkeiten im Software-Engineering darstellt (Balzert, 2011, S. 26; Balzert & Balzert, 2009, S.29; Reuter, Knietzsch et al., 2019).

Das durchgeführte Experiment sollte dabei die Frage beantworten, inwiefern AR die Fähigkeit des Abstrahierens beeinflusst und damit Abstraktes Denken unterstützt.

Abstraktion wird dabei definiert als, „Verallgemeinerung, das Absehen vom Besonderen und Einzelnen, das Loslösen vom Dinglichen. Abstraktion ist das Gegenteil von Konkretisierung“ (Balzert & Balzert, 2009, S.26). Anstelle von Abstraktion wird häufig auch von Modellbildung gesprochen. Durch die Tätigkeit des Abstrahierens vom Konkreten wird ein Modell der realen Welt erstellt, „d. h. das Modell repräsentiert die reale Welt durch sein charakteristisches Verhalten“ (Balzert & Balzert, 2009, S.27). Kramer (2007) betrachtet Modellierung und damit Abstraktion als die wichtigsten Tätigkeit eines Ingenieurs: „Modeling is the most important engineering technique; models help us to understand and analyze large and complex problems“ (Kramer, 2007, S. 41).

Innerhalb eines Software Engineering-Kurses durchlaufen die Studierenden alle Phasen der Softwareentwicklung von einer Produktidee zu konkreten Anforderungen über einen Softwareentwurf bis hin zum Softwareprodukt. Die Studierenden müssen die „ständigen Wechsel zwischen *Abstrahieren* und *Konkretisieren*“ (Balzert & Balzert, 2009, S. 29) verstehen. Am Anfang einer Softwareentwicklung steht in der Regel ein Abstraktionsprozess. Sobald ein Modell erstellt wurde, ist „dieses der Ausgangspunkt für die Konkretisierung“ (Balzert & Balzert, 2009, S. 30), d. h. das Modell wird „bis auf die Ebene der verwendeten Programmiersprache konkretisiert“ (Balzert & Balzert, 2009, S. 30).

„In der objektorientierten Softwareentwicklung lassen sich prinzipiell folgende drei Abstraktionsebenen unterschieden (Hesse, Barkow, von Braun, Kittlaus & Scheschonk, 1994):

- Die Exemplar-Ebene,
- die Typ-Ebene und
- die Meta-Ebene, genauer gesagt die Meta-Typen-Ebene“ (Balzert & Balzert, 2009, S. 30).

Alle drei Ebenen können eine Rolle in der Objektorientierung spielen.

Weiterhin lässt sich unterscheiden zwischen eindimensionaler und mehrdimensionaler Abstraktion, wobei folgende Dimensionen bzw. Sichten eine Rolle spielen (Balzert & Balzert, 2009; Pennington, Lee & Rehder, 1995; Reuter, Knietzsch et al., 2019; Rumbaugh, Eddy, Blaha & Premerlani, 1991):

- Statik
- Dynamik
- Logik

Während eindimensionale Abstraktion beispielsweise beim Konzept des Zustandsautomaten eintritt, ist mehrdimensionale Abstraktion bei Klassen (Daten und Funktionen) nötig. Die OOA berücksichtigt sogar vier verschiedene Dimensionen: Statik mit Daten und Funktionen, Dynamik, Logik (Balzert & Balzert, 2009, S. 33). Je mehr Dimensionen berücksichtigt werden müssen, desto schwieriger ist die Wahl der geeigneten Abstraktion (Balzert & Balzert, 2009). Studierende müssen demnach die verschiedenen Dimensionen kennen und geeignete Abstraktionen wählen können, um Modelle einerseits zu erstellen und andererseits zu verarbeiten.

Abbildung 6.44: Objektansicht

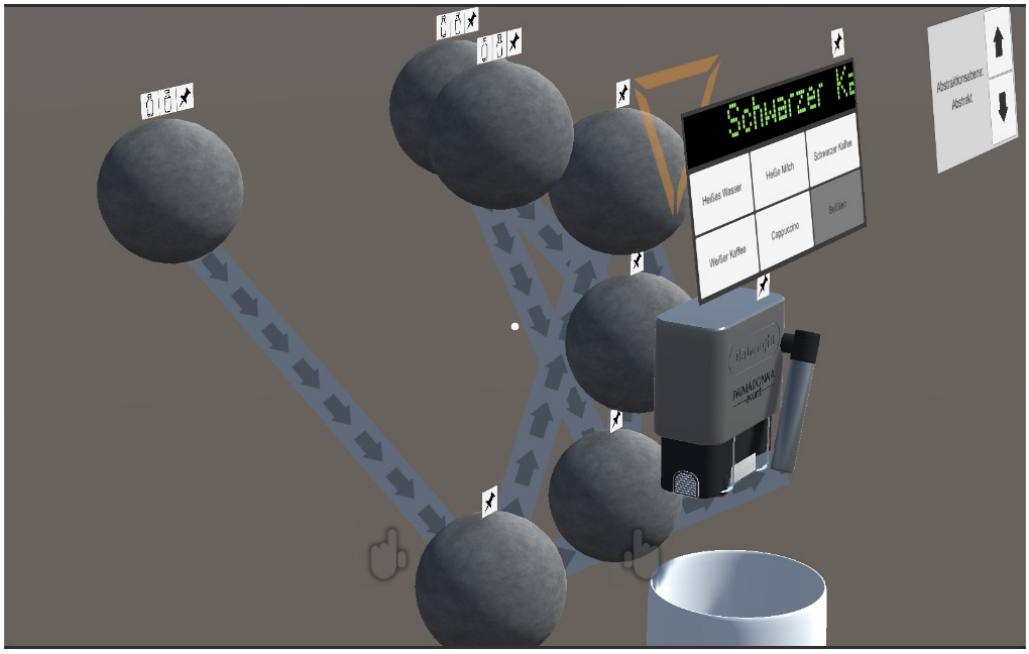


Abbildung 6.45: Dynamische Ansicht

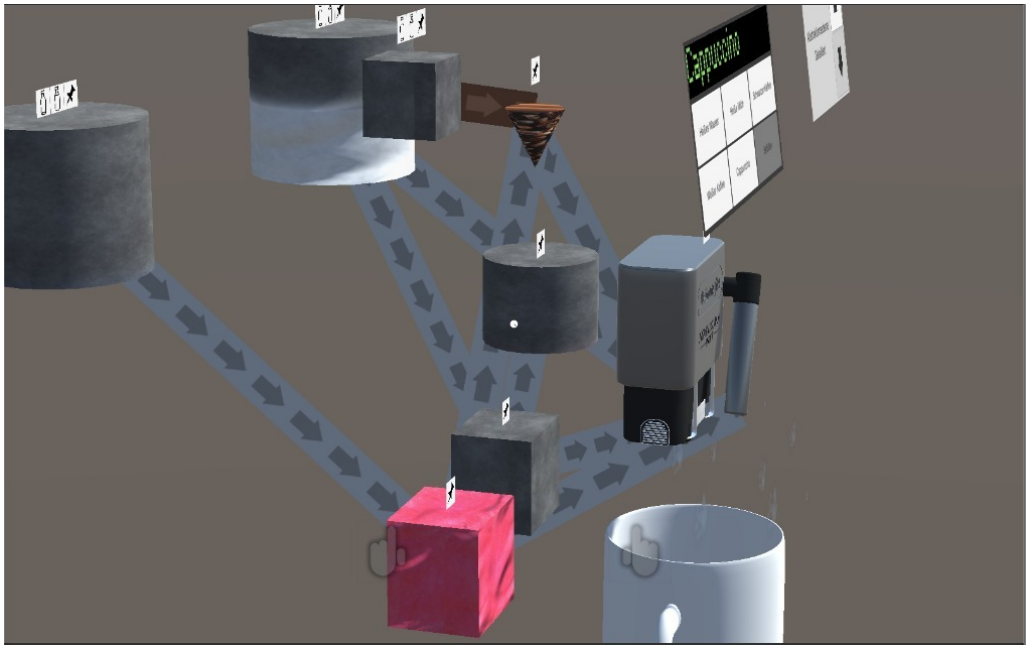


Abbildung 6.46: Funktionsansicht

Auf dieser Basis wurde ein Unterrichtsszenario konzipiert, in dem die Studierenden eine Kaffeemaschine entwerfen müssen. Das Szenario wird durch eine AR-Anwendung unterstützt (siehe Abb. 6.44, Abb. 6.45, Abb. 6.46), die es den Studierenden ermöglicht, eine interaktive Ansicht des betreffenden Systems (der Kaffeemaschine) auf den verschiedenen Abstraktionsebenen abzurufen, wie von Balzert und Balzert (2009), Pennington et al. (1995) oder Rumbaugh et al. (1991) vorgeschlagen. Die AR-Anwendung soll Studierende für die Bedeutung der verschiedenen Dimensionen (die drei Schichten) für die Software-Entwicklung sensibilisieren (Reuter, Knietzsch et al., 2019).

Das Studiendesign war folgendermaßen angelegt:

- Within-Subject-Design: Das bedeutet, dieselbe Person absolviert nacheinander alle experimentellen Bedingungen.
- Kontrolliertes Laborexperiment: Das bedeutet, es herrschen identische kontrollierte Bedingung für alle Teilnehmer.
- Querschnittsorientiert: Das Experiment findet einmalig statt.
- Es werden Audio-Aufnahmen erstellt (Reuter, Knietzsch et al., 2019).

Die Aufgabenstellung hat eine dreigliedrige Struktur und ähnelt der von Kramer (2006) und Hazzan und Kramer (2016) Vorgeschlagenen:

Studierende (N=9) mussten einzeln die Rolle des Erfinders einer Kaffeemaschine übernehmen:

> „Die Studierenden werden gebeten, jemand anderem ein Thema X (ein System, eine Maschine usw.) zu erklären. Sie werden

> gebeten, zu beschreiben, wie sie X in zwei Fällen erklären würden: wenn die Person, der sie es erklären, X sehen kann und wenn sie X nicht sehen kann. Sie werden auch gebeten, die Überlegungen zu erläutern, die der Formulierung jeder Beschreibung zugrunde lagen, und zu erklären, wie sich diese Formulierungen zu den verschiedenen Abstraktionsebenen verhalten“ (Kramer, 2006, Muster 5; Reuter, Knietzsch et al., 2019).

Im ersten Schritt sollten die Studierenden einem Entwickler die Kaffeemaschine anhand einer Feature-Liste erklären. Im zweiten Schritt sollten sie die Kaffeemaschine einem technisch unerfahrenen Kunden erklären (Reuter, Knietzsch et al., 2019). Schließlich wurde den Studierenden die AR-Applikation zur Verfügung gestellt. Nachdem sie sich mit der Anwendung und der AR-Brille vertraut gemacht hatten, wurden die Studierenden (dritter Schritt) gefragt, welche Änderungen sie an ihren Erklärungen gegenüber den vorherigen Rollen (Kunde und Softwareentwickler) vornehmen würden (Reuter, Knietzsch et al., 2019).

6.10.2.1 *Bewertung*

Die Auswertung des Experiments erfolgte mit leitfadengestützten Interviews zur Nützlichkeit und Anwendbarkeit für den Unterricht aus (siehe Tabelle 6.6 Tabelle 6.8 und Tabelle 6.10). Zudem wurden die erstellten Notizen der Teilnehmer ausgewertet. Die Daten wurden mit Codes abgeleitet aus den Leitfragen markiert, die die Aussagen der Teilnehmer kategorisieren (Reuter, Knietzsch et al., 2019).

Nachfolgend findet sich eine Liste der Leitfragen (Reuter, Knietzsch et al., 2019):

- Würden Sie weitere Dokumente für Ihre Erklärung verwenden?
- Welche Abstraktionsebenen würden Sie verwenden, um verschiedene Arten von Diagrammen zu modellieren? (Sequenzdiagramm, Klassendiagramm, Use-Case-Diagramm, Zustandsautomaten)
- Was denken Sie über die Nutzbarkeit der AR-Brille/Anwendung?
- Haben Sie das Gefühl, dass Sie aus der Anwendung etwas gelernt haben?
- Halten Sie es für sinnvoll, die Anwendung in der Vorlesung einzusetzen?

Für Phase eins wurde jede Zuordnung eines Merkmals zu einer Erklärung mit einem Code gekennzeichnet. Daraus ergab sich eine Tabelle mit Antworten auf einzelne Fragen/Aufgaben. Diese Tabelle wurde mit den

erwarteten Antworten verglichen. Für jede richtige Zuordnung wurde ein Punkt vergeben. Insgesamt konnten 56 Punkte für eine völlig korrekte Erklärung für den Kunden und weitere 56 Punkte für die Erklärung für den Software-Entwickler erreicht werden (Reuter, Knietzsch et al., 2019).

Die Antworten auf die Fragen in Phase drei wurden in Konzepte kategorisiert, über die die Teilnehmer sprachen. Zum Beispiel sagte ein Teilnehmer, dass es mehr Spaß machen würde, AR zu verwenden, als reine Textbeschreibung, während ein anderer Teilnehmer sagte, dass es interessant sei, AR zu verwenden. Beide wurden dann in „höhere Motivation" kategorisiert (Reuter, Knietzsch et al., 2019).

6.10.2.2 *Ergebnisse*

Die Ergebnisse zeigen keine Veränderung des *Abstraktionsverhaltens* nach Anwendung der Applikation. Die Teilnehmer erzielten im Durchschnitt hohe Punktzahlen in Phase eins und hatten im Allgemeinen wenig bis keine Änderungen, die sie in Phase drei vornehmen wollten (Reuter, Knietzsch et al., 2019).

Wie aus Tabelle 6.9 und Tabelle 6.5 ersichtlich ist, lag die durchschnittliche Punktzahl bei fast 80%. Dieser Durchschnittswert ist sogar noch höher, wenn „Grauzonen" berücksichtigt werden: Zwei Einträge in Tabelle 6.9 sind markiert: Das liegt daran, dass diese beiden Teilnehmer eindeutig nicht von einem Software-Entwickler, sondern von einem Hardware-Entwickler sprachen. Um dieser Tatsache Rechnung zu tragen, werden zwei Durchschnittswerte berechnet. Einer mit und einer ohne die genannten Teilnehmer. Beide Durchschnittswerte unterscheiden sich nur geringfügig. Am Ende ihres jeweiligen Experiments wurden die unterstrichenen Teilnehmer ausdrücklich gefragt, was sie ändern würden, wenn der Entwickler sich nur um die Software kümmern würde. Beide antworteten, dass Merkmale wie Farbschemata oder die Art des Stahls, aus dem die Rohre hergestellt sind, irrelevant seien. Dies deutet darauf hin, dass sie die ursprüngliche Aufgabe nicht verstanden haben, aber dennoch ein gutes Verständnis für die damit verbundenen Konzepte hatten. Es ist möglich, dass mehrere Teilnehmer als diese beiden mit der falschen Prämisse begonnen haben, dies aber nicht sichtbar wurde (Reuter, Knietzsch et al., 2019).

Das Feedback der Teilnehmer war überwiegend positiv, wie aus den Tabellen (Tabelle 6.6, Tabelle 6.8 und Tabelle 6.10) hervorgeht. Insgesamt waren zwei der neun Teilnehmer der Meinung, dass die gezeigte Anwendung nicht sinnvoll war. Die qualitative Auswertung der Interviewergebnisse zeigt weiterhin, dass die Studierenden die Notwendigkeit und Bedeutung verschiedener Perspektiven erkennen, womit das Ziel

Tabelle 6.4: Erzielte Punkte der Teilnehmer in der ersten Phase der Aufgabe (Erläuterung Software-Entwickler)

#Teilnehmer	Punkte	Anteil(%)
TN 1	47	83,93%
TN 2	49	87,50%
TN 3	42	75,00%
TN 4	45	80,36%
TN 5	46	82,14%
TN 6	44	78,57%
TN 7	43	76,79%
TN 8	39	69,64%
TN 9	40	71,43%
Durchschnitt ohne 6, 8	44,29	79,08%
Durchschnitt	43,89	78,37%

der Studie erfüllt wird. Die Studierenden finden es gut, zwischen den Abstraktionsebenen wechseln zu können und die Möglichkeit zu haben, die Maschine zu zerlegen (Reuter, Knietzsch et al., 2019).

6.10.2.3 *Einschränkungen & Methodische Grenzen*

Die Studie weist folgende Einschränkungen auf:

- Die Stichprobe ist mit $N = 9$ zu klein, um signifikante Ergebnisse zu erhalten und eine verallgemeinernde Schlussfolgerung zu ziehen.
- Die Studierenden waren Novizen in der Modellierung. Die Konfrontation mit einem neuen Werkzeug, einer neuen Technologie und einem neuen Lerninhalt kann zu Überforderung und kognitiver Überlastung führen.
- Zielgruppe waren Studierende der Nicht-Hauptfachrichtung Informatik. Bei Studierenden der Informatik können die Ergebnisse anders ausfallen.
- Unklarheiten zwischen den Rollen Softwareentwickler und Hardwareentwickler waren in dieser Studie essentiell und könnten die Reliabilität der Ergebnisse beeinträchtigen.

Tabelle 6.5: Erzielte Punkte der Teilnehmer in der zweiten Phase der Aufgabe (Erläuterung Kunde)

#Teilnehmer	Punkte	Anteil(%)
TN 1	42	75,00%
TN 2	39	69,64%
TN 3	43	76,79%
TN 4	50	89,29%
TN 5	44	78,57%
TN 6	45	80,36%
TN 7	45	80,36%
TN 8	42	75,00%
TN 9	46	82,14%
Durchschnitt	44	78,57%

Tabelle 6.6: Nummerische Darstellung der Ergebnisse zur ersten Frage der Evaluation: „Halten Sie es für sinnvoll, die Anwendung in der Vorlesung einzusetzen?“

Feedback	Anzahl
Ja (keine weitere Erklärung gegeben)	3
Nicht nützlich	2
Gut, um verschiedene Perspektiven zu sehen	2
Schwerfällig	2
Guter Überblick	2
Interessantes/Motivierendes	2
Hohe Entwicklungskosten	1
Gut für das räumliche Verständnis	1
Insgesamt positiv	7
Gesamt Negativ	2

Tabelle 6.7: Nummerische Darstellung der Ergebnisse zur zweiten Frage der Evaluation: „Welche Diagrammart würden Sie für welches Abstraktionslevelnutzen?“

	Ausführlich	Abstrakt	Top-Level	Unentschieden
Sequenzdiagramm	4	3	0	2
Klassendiagramm	5	5	0	1
Use-Case-Diagramm	0	1	8	0
Zustandsdiagramm	6	3	0	0

Tabelle 6.8: Nummerische Darstellung der Ergebnisse zur dritten Frage der Evaluation: „Haben Sie das Gefühl, dass Sie aus der Anwendung etwas gelernthaben?“

Feedback	Anzahl
Über AR	3
Über Automaten	2
Über Abstraktion	2
Neutral	2
Gesamt Positiv	7
Gesamt Negativ	1

Tabelle 6.9: Erzielte Punkte der Teilnehmer in der ersten Phase der Aufgabe (Erläuterung Software-Entwickler)

#Teilnehmer	Punkte	Anteil(%)
TN 1	47	83,93%
TN 2	49	87,50%
TN 3	42	75,00%
TN 4	45	80,36%
TN 5	46	82,14%
TN 6	44	78,57%
TN 7	43	76,79%
TN 8	39	69,64%
TN 9	40	71,43%
Durchschnitt ohne 6, 8	44,29	79,08%
Durchschnitt	43,89	78,37%

6.10.3 *Zusammenfassung zum Einsatz von Augmented Reality-Applikationen*

Aus beiden AR-Experimenten können aufgrund ihrer geringen Stichprobe keine allgemeingültigen Erkenntnisse abgeleitet werden. Zudem wurden nur kontrollierte Experimente durchgeführt, es können damit keine Aussagen über AR im Feldeinsatz getroffen werden. Die festgestellten Indizien zeigen, dass Studierende mit der Anwendung von AR motivierter sind, und gleich gut abschneiden wie Studierende, die keine AR Applikation eingesetzt haben. Im Einsatz von AR Applikationen für alle evaluierten Fälle, konnten gesteigerte Motivation, gleichbleibende Performance (Leistung) und Unterstützung bzw. Sensibilisierung bei Abstraktion bzw. Konkretisierung und damit Visualisierung unsichtbarer Konzepte nachgewiesen werden. Damit wird das Potenzial dieser Technologie für zukünftige Forschung im Bereich der Hochschullehre aufgezeigt. Die tatsächliche Eignung muss jedoch vor allem in größer angelegten Studien noch nachgewiesen werden, weshalb AR im Rahmen der Arbeit nicht weiter fortgeführt wird.

Tabelle 6.10: Nummerische Darstellung der Ergebnisse zur vierten Frage der Evaluation: „Was halten Sie von der Nutzbarkeit der AR-Brille und derAnwendung?“

Feedback	Anzahl
Intuitiv	6
Funktionelle	5
Schleppend	1
Zu schwer	1
Besser als VR	1
Schlecht im Sonnenlicht	1
Gesamt Positiv	8
Gesamt Negativ	1

6.11 EYE-MOVEMENT MODELING EXAMPLE

Das dritte Experiment in der Reihe der Evaluation der experimentell technologiebasierter Scaffolds, untersucht den Einsatz von Eye-Movement Modeling Examples (EMME). Wie bereits im einführenden Abschnitt 6.1.2.2 zu EMMEs beschrieben, sind diese eine Form der *„worked examples“* (Arnold et al., 2017; Chandler et al., 2001; Franke-Braun, 2004) und stellen eine besondere Art der videobasierten Modellierung dar (Jarodzka et al., 2012, S.815). Sie zeigen einen Experten, der eine Aufgabe bearbeitet und dazu erläutert, wie er die Aufgabe ausführt, während gleichzeitig seine Augenbewegungen visualisiert werden. Diese Kombination erfüllt den Wunsch der Studierenden nach einem „Livebeispiel“, in dem die Gedanken des Dozenten besonders deutlich werden (Jarodzka et al., 2012; Mason et al., 2015; Sweller, 1988).

Der Grundgedanke der durchgeführten Studie bestand darin, zu evaluieren, inwiefern ein EMME Studierenden als Scaffold während der Modellierung dienen kann. Da die Modellierung von Sequenzdiagrammen in Item 1 der Fragebogenstudie (siehe Abschnitt 5.3.2 bzw. Abschnitt 5.3.3.1) von Studierenden als schwierigste zu modellierende Diagrammart bewertet wurden, wird im Rahmen der Studie der Einsatz des EMMEs zum Thema Sequenzdiagrammen evaluiert. Die Arbeit wurde gemeinsam mit zwei Masteranden durchgeführt (Schreistetter, 2021; Stark, 2021).

6.11.1 *Methodik*

Die Studie ist als Eyetracking-Studie konzipiert. Das Design der Studie ist ein 2x2 faktorielles Design mit „Bedingung“ (experimentelle Bedingung (Experimentalgruppe) vs. Kontrollbedingung (Kontrollgruppe)) als Between-Subject-Faktor und Schwierigkeit (Einfache und Komplex) als Within-Subject-Faktor. Für den Faktor „Bedingung“ werden alle Teilnehmer entweder der experimentellen Bedingung oder der Kontrollbedingung zugeordnet. Teilnehmer der Kontrollgruppe erhalten ein Video, in welchem der UML-Modellierungsprozess von einem Modell durchgeführt wird, das nur verbale Erklärungen gibt, während die Experimentalgruppe dasselbe Video erhält, in dem der Blick des Modells durch einen beweglichen Lichtkegel dargestellt wird (EMME-Video). Für den Faktor Schwierigkeit durchlaufen alle Teilnehmer zunächst eine Lernphase auf leichtem Niveau und dann eine Lernphase auf schwierigem Niveau.

6.11.1.1 *Ziel*

Ziel der Studie ist es zu evaluieren, ob die Teilnehmer der Versuchsgruppe . . .

- bessere Performanz (Leistung) aufweisen, als die Teilnehmer der Kontrollgruppe.
- geringere kognitive Belastung während des Lernens aufweisen, als die Teilnehmer der Kontrollgruppe (also kognitiv nicht überbelastet werden).
- dem Blick des Modells folgen.
- EMMEs als Lehrmaterial positiv bewerten.

6.11.1.2 *Zielgruppe*

Die Zielgruppe sind Studierende, welche eine Vorlesung zum Thema UML besuchen oder ähnliche Veranstaltungen oder diese Veranstaltungen bereits besucht haben. Die Durchführung findet im Sommersemester 2020 statt. Aufgrund der möglichen unterschiedlichen Hintergründe der Studierenden ist davon auszugehen, dass deren Vorwissen bezüglich UML unterschiedlich ist. Um potentielle Einflüsse des Vorwissens auf die Wirksamkeit der EMMEs feststellen zu können, wird das Vorwissen anhand einer Vorwissenserhebung gemessen und Ergebnisse der Vorwissenserhebung werden in der Auswertung berücksichtigt.

6.11.1.3 *Vorgehensweise*

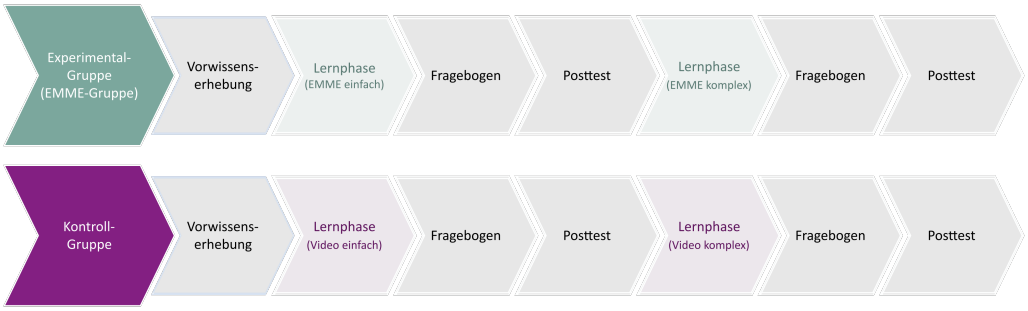


Abbildung 6.47: Ablauf der Studie

Der Ablauf der Studie war wie folgt (Abb. 6.47): In beiden Gruppen (Kontrollgruppe und Experimentalgruppe) wurde eine Vorwissenserhebung durchgeführt. Im Anschluss daran startete der Experiment-Anteil, der Eyetracking beinhaltete. Zunächst sahen die Teilnehmer abhängig von ihrer Gruppe für fünf Sekunden einen Baseline-Stimulus, um den Pupillendurchmesser ohne Belastung zu erfassen (siehe Abschnitt 6.11.2.3). Anschließend startete, abhängig von der Gruppenzuordnung ein Video, das für die Teilnehmer der Kontrollgruppe Audio- und Bildschirmaufnahme des Modells enthielt und für die Teilnehmer der Versuchsgruppe das identische Video mit Überlagerung der Blickbewegungen des Modells. Dieser Schritt entsprach der ersten Lernphase (farbliche Hervorhebungen in Abb. 6.47). Nach dem Video, beantworteten die Teilnehmer einen Fragebogen (siehe Abschnitt 6.11.2.7). Anschließend zu diesem Schritt erfolgte in beiden Gruppen der Posttest und die Teilnehmer modellierten in der Modellierungsumgebung, die auch in den Videos verwendet wurde, eine anderes Szenario, als das in den Videos gezeigte. Zuvor erhielten sie die Aufgabentexte zum Lesen. Dieser Schritt beendete den ersten Teil der Erhebung. Der zweite Teil startete wiederum mit einer Lernphase und es wurde ein komplexeres Modell erstellt. Die weiteren Schritte blieben identisch. Die Aufgabenbeschreibung für beide Posttests unterschied sich nicht, die Aufgabenstellung unterschied sich in ihrer Komplexität.

6.11.2 *Instrumente*

Folgende Materialien bzw. Instrumente wurden für die Erhebung ausgewählt, konzipiert und eingesetzt.

6.11.2.1 *Eingesetzte Hardware und Software*

Für die Eyetracking Studie wurden vier Eyetracker des Typs Tobii Spectrum eingesetzt. Die Studie wurde mittels Tobii Pro Lab Version 1.145 designed und aufgezeichnet. Die Modellierungsumgebung ist wie auch in den vorherigen Experimenten zu AR, Ariadne 2D. Diese wurde im Rahmen dieser Arbeit konzipiert und realisiert. Im zeitlichen Verlauf der vorliegenden Arbeit fand das Experiment mit dem EMME später statt, sodass eine neuere Version von Ariadne 2D eingesetzt wurde. Für den Online-Fragebogen wurde Limesurvey[34] eingesetzt.

6.11.2.2 *Vorwissens-Erhebung*

Um den anfänglichen Wissensstand der Studierenden auf dem Gebiet der UML, insbesondere der UML-Sequenzdiagramme, zu erfassen, wurde ein Pretest mit acht Items konzipiert (siehe Anhang A.6). Es gibt vier Items, bei denen die Studierenden entscheiden müssen, ob eine Aussage über UML-Sequenzdiagramme richtig oder falsch ist, und vier offen gestaltete Items, bei denen die Studierenden bestimmte Komponenten von Sequenzdiagrammen modellieren müssen. Um dieses Verfahren effizient zu gestalten, wird der Pretest nicht am PC, sondern auf Papier durchgeführt. Im Vorwissenstest können die Teilnehmer max. 15 Punkte erreichen.

6.11.2.3 *Pupillendurchmesser*

Studien zufolge, lässt die Größe der Pupille Rückschlüsse auf die kognitive Belastung eines Teilnehmers zu.

Zur Berechnung der hypothetischen Pupillenerweiterung ist es notwendig, den Basis-Pupillendurchmesser der Teilnehmer zu erfassen. Dies geschieht, indem dem Teilnehmer für 5 Sekunden ein Stimulus gezeigt wird, der keinerlei Informationen enthält, um sicherzustellen, dass der Stimulus keine kognitive Belastung für den Empfänger erzeugt (siehe Holmqvist et al., 2013, S. 393). Da die Pupillenerweiterung durch die Leuchtdichte eines visuellen Reizes beeinflusst wird, ist es notwendig, die Leuchtdichte zu kontrollieren. Die Stimuli für die Experimentalgruppe (EMME-Video) und für die Kontrollgruppe (normales Video) unterscheiden sich in ihrer Leuchtdichte, da das EMME-Video durch einen Lichtkegel gekennzeichnet ist, der den Blick des Modells darstellt und von einem abgedunkelten und unscharfen Bereich umgeben ist. Im Gegensatz dazu erhält die Kontrollgruppe ein unbearbeitetes Video, bei

34 https://www.limesurvey.org/de/

dem es sich um eine Bildschirmaufzeichnung handelt, die aus einer Modellierungsumgebung mit hellem/weißen Hintergrund besteht. Daher erhielt die experimentelle Gruppe einen Baseline-Stimulus, der auf der Grundlage des EMME-Videos erzeugt wurde (Abb. 6.48), und die Kontrollgruppe erhielt einen Baseline-Stimulus, der auf der Grundlage des Kontrollvideos erzeugt wurde (Abb. 6.49). Mit dessen Hilfe gelingt daher eine interindividuelle Kontrolle der physiologischen Gegebenheiten der Teilnehmer (siehe z. B. Holmqvist et al., 2013). Der Stimulus ist direkt sichtbar, bevor sich der Teilnehmer das erste Video in der Lernphase ansieht, daher wird er auf der Basis von Bildern aus dem jeweiligen „Einfachen“ Video erzeugt.

Abbildung 6.48: Baseline Stimulus für die Experimentalgruppe

Abbildung 6.49: Baseline Stimulus für die Kontrollgruppe

6.11.2.4 *Aufgaben*

Um einen Wissenstransfer von Lern- auf Testphase sicherzustellen, gab es zwei verschiedene Szenarien (Lernphase: Szenario 1 bzw. Aufgaben-

beschreibung 1, Testphase: Szenario 2 bzw. Aufgabenbeschreibung 2). Die Lernphase spielte jeweils im gleichen Szenario 1, aus welchem eine einfache und eine komplexere Aufgabenstellung extrahiert wurden, welche vom Modell während des Videos bearbeitet wurde. Die Testphase spielte auch jeweils im gleichen, jedoch anderen, Szenario 2, aus welchem auch eine einfache und eine komplexe Aufgabe extrahiert wurde. Die beiden Aufgabenbeschreibungen erfolgten szenariobasiert, d. h. es wurde eine Problemstellung in natürlicher Sprache beschrieben. Um ein korrektes Sequenzdiagramm (oder ein anderes UML-Diagramm) zu erstellen, mussten die für die Software erforderlichen Elemente aus dieser Textbeschreibung extrahiert werden. Es wurden demnach zwei Aufgaben konstruiert, eine für das Modell des EMMEs und eine für die Studierenden. Beide Aufgaben wurden so formuliert, dass sie geeignete Passagen für beide Phasen (einfach und komplex) enthalten.

6.11.2.5 *Lernphase - Videos*

Den Teilnehmern wurden zwei verschiedene Videos zur Verfügung gestellt, in denen ein Experte UML-Diagramme erstellte und sein Vorgehen verbal erläuterte. Die Videos sind Screencast-Videos des Modellierungsprozesses, die von den Studierenden nicht angehalten werden können. Die Teilnehmer saßen während der Lernphase vor Eyetrackern.

Der Experte (i.e. das Modell des EMMEs) führte sowohl bei Videos auf einfacher als auch auf schwieriger Ebene die Erstellung eines UML-Sequenzdiagramms durch. Im Idealfall liegen, bevor Sequenzdiagramme für einen Softwaresystem entworfen werden, bereits Use-Case-Beschreibungen und Klassendiagramme für das Softwaresystem vor (siehe Abschnitt 4.3).

Deshalb wurde für die Aufnahme des EMMEs aus der Aufgabenbeschreibung eine Use-Case-Beschreibung und ein Klassendiagramm im Grobentwurf (ohne Attribute und Methoden) extrahiert. Diese wurden im EMME-Video auf der linken Seite des Bildschirms präsentiert. Auf der rechten Seite wurde die Modellierungsumgebung gezeigt.

Das Verhalten des Modells kann als eine Mischung aus natürlich und didaktisch charakterisiert werden. Für die Aufzeichnung des Modellierungsprozesses als EMME wurde die erste Aufnahme gewählt, d. h. das Modell erstellte das Diagramm zum ersten Mal und kannte den Inhalt der textuellen Beschreibung und des Klassendiagramms nicht. Zusätzlich wurde das Modell gebeten, wichtige Aspekte hervorzuheben und zusätzliche Erklärungen so zu geben, dass Studierende sie verstehen können. Das Modell war eine Dozierende im Bereich Software Engineering. Die Entscheidung für dieses Modell zielte auf einen kongruenten Stil der

Erklärungen in den Lektionen und in den Videos ab. Der Versuchsgruppe wurden die gleichen Videos zur Verfügung gestellt wie der Kontrollgruppe. Der einzige Unterschied zum Video für die Kontrollgruppe bestand darin, dass der Blick des Modells zusätzlich auf das Video überlagert wurde[35] (Abb. 6.50, Abb. 6.51). Aus früheren Studien ist bekannt, dass das Anstrahlen des Blicks, anstelle der Darstellung des Blicks durch einen farbigen Punkt, für das Lernen und die Anleitung effektiver ist (Jarodzka et al., 2012). Daher wurde der Blick durch einen Lichtkegel der nicht fokussierten Bereiche hervorgehoben. Laut Litchfield, Ball, Donovan, Manning und Crawford (2010) treten Leistungsverbesserungen nur dann ein, wenn die Modelle tatsächlich Suchverhalten zeigen (Litchfield et al., 2010). Deshalb wurde auch die erste Aufnahme des Modells gewählt. Das Augenbewegungsmodell verhielt sich also didaktisch, während sein Blick aufgezeichnet wurde. Das heißt, das Modell wurde angewiesen, möglichst integratives Verhalten zu „nutzen", z. B. offensichtliche Zusammenhänge zwischen entsprechenden Elementen in Text und Diagramm oder beiden Diagrammen herzustellen.

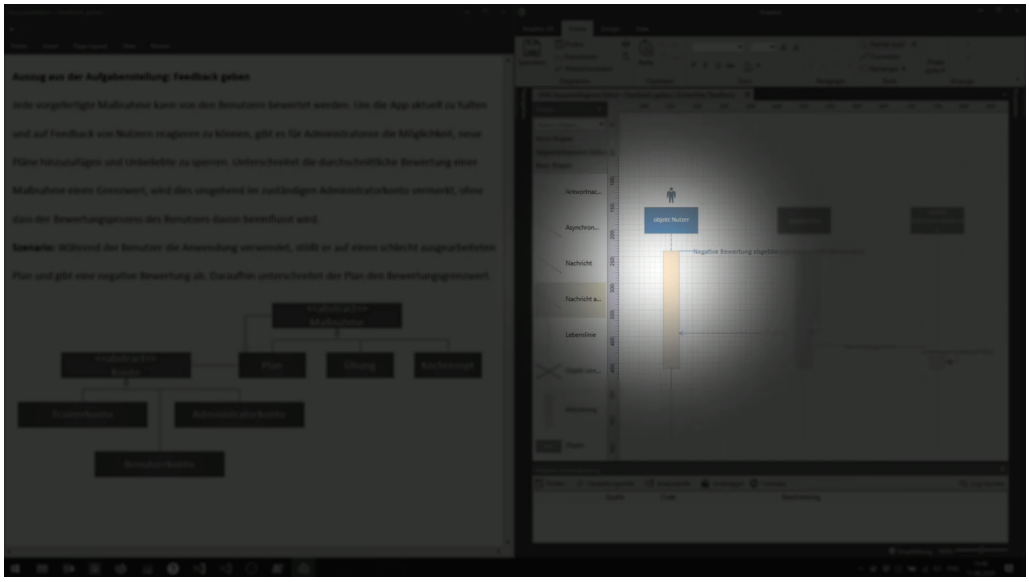


Abbildung 6.50: Screenshot aus dem Experimentsetting der Experimentalgruppe

35 https://tinyurl.com/EmmeEasy, https://tinyurl.com/EmmeHard

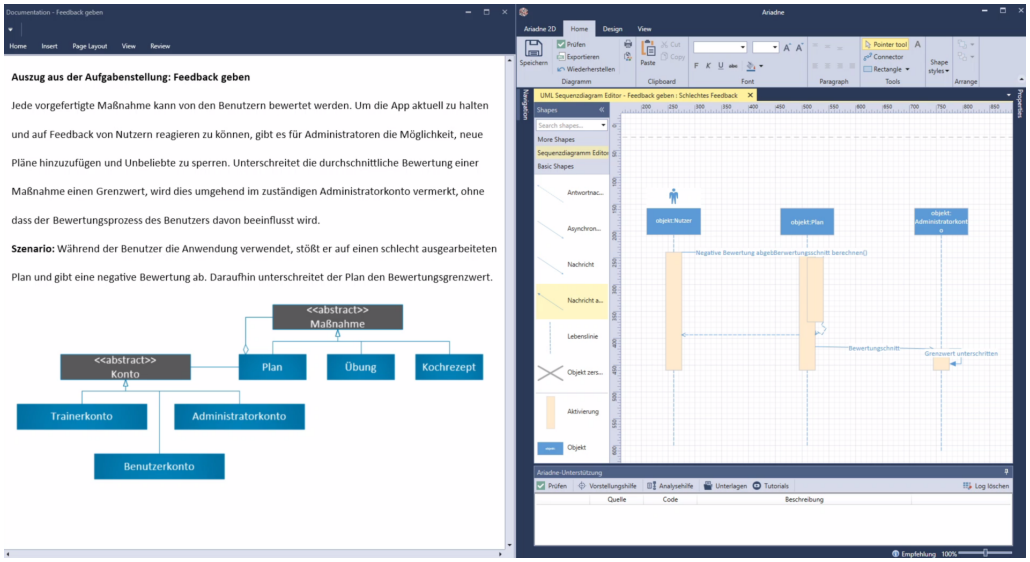


Abbildung 6.51: Screenshot aus dem Experimentsetting der Kontrollgruppe

6.11.2.6 *Posttest*

Ziel der Posttests war es zu überprüfen, ob die EMMEs einen Einfluss auf die Leistungen der Studierenden hatten. Deshalb wurden die Studierenden nach Lernphase 1 und nach Lernphase 2 gebeten ein Sequenzdiagramm modellieren. Im ersten Posttest mussten die Studierenden ein einfaches Sequenzdiagramm modellieren. Im zweiten Posttest war analog dazu ein komplexeres Diagramm gefragt. Ein einfaches Diagramm wurde angenommen als gering in Umfang und Komplexität. Zudem enthielt es nur die Elemente, Objekt, Lebenslinie, Aktivierungslinie, synchrone Nachricht und asynchrone Nachricht. Ein komplexes Diagramm war größer im Umfang und enthielt einen größeren Elementumfang. Zu den bereits beschriebenen Elementen wurden nun auch Fragmente erwartet.

Zur Auswertung des Posttests wurde ein umfassendes Bewertungsschema entwickelt, das Qualitätsaspekte zur Erstellung von Sequenzdiagrammen berücksichtigt (siehe Schreistetter (2021) und Anhang A.7). Es basiert auf dem Ansatz von Schmedding und Vasileva (2017), Arendt, Mantz und Taentzer (2009) und Unhelkar (2005). Je zwei Reviewer bewerteten die erstellten Diagramme anhand des Bewertungsschemas, anschließend wurden die Bewertungen der beiden Reviewer verglichen und im Falle von Bewertungsdifferenzen wurden die Aspekte der Bewertung, die Differenzen erzeugten, neu bewertet. Im Posttest konnten die Teilnehmer max. 78 Punkte erreichen.

6.11.2.7 *Fragebogen*

Um etwas über die Einstellung und Akzeptanz der Studierenden gegenüber den EMME-Videos herauszufinden, werden in einem Online-Fragebogen die folgenden Fragen gestellt:

- Auf einer Skala von 1 (überhaupt nicht) bis 5 (sehr gut) - Wie hat Ihnen das Lehrmaterial gefallen?
- Auf einer Skala von 1 (sehr hinderlich) bis 5 (sehr unterstützend) - Wie würden Sie die Unterstützung durch das Lehrmaterial bewerten?
- Auf einer Skala von 1 (sehr einfach) bis 5 (sehr schwierig) - Als wie schwierig empfanden sie das Diagramm?
- Wenn es noch etwas gibt, was Sie uns über das Lehrmaterial mitteilen möchten, können Sie das gern hier tun:
- Ist Ihnen während des Betrachtens des Lehrmaterials etwas im Verhalten des Modells (Person, die die Aufgabe im Video bearbeitet) aufgefallen?

6.11.3 *Auswertung*

Im Folgenden werden erste Ergebnisse der Studie vorgestellt. Die Auswertung erfolgt zunächst unter Einbezug der oben genannten Ziele. Eine detaillierte Beschreibung der Auswertungsschritte, insbesondere zur Datenaufbereitung, findet sich in Stark (2021) und Schreistetter (2021).

Insgesamt konnten $N = 24$ Teilnehmer rekrutiert werden, die randomisiert der Experimentalgruppe ((EG), $N = 12$) und Kontrollgruppe ((KG), $N = 12$) zugeteilt wurde.

Nachfolgend sind zentrale Hypothesen zusammengefasst, die sich mit Hilfe der Daten bestätigen lassen:

1. Die Abweichung des Blickes der EMME-Gruppe (EG) vom Blick des Modells ist geringer als die Abweichung des Blickes der Kontrollgruppe vom Blick des Modells.
2. Die Experimentalgruppe bewertet das Lehrmaterial positiver als die Kontrollgruppe.
3. Es gibt eine größere Steigerung im kognitiven Load bei der Kontrollgruppe.
4. Teilnehmer der EMME Gruppe zeigen bessere Leistung im Posttest als Teilnehmer der Kontrollgruppe (KG)
5. Das Vorwissen beeinflusst die Performanz der Teilnehmer im Posttest in den verschiedenen Schwierigkeitsstufen.

6.11.3.1 *Erkenntnisse zur ersten Hypothese*

Ziel der ersten Hypothese ist die Überprüfung, ob die EMME-Gruppe (EG) dem Lichtkegel auch tatsächlich folgt. Um diese Hypothese zu überprüfen, d. h., ob die Teilnehmer der EG dem Blick des Modells folgen, wird zu jedem Zeitpunkt (t_{n}) während der Lernphase die „Nähe" des Blickpunktes des Teilnehmers zum Blickpunkt des Modells berechnet. Der Blickpunkt ist ein einzelner Punkt auf dem Bildschirm, welcher sich aus einer X- und einer Y-Koordinate zusammensetzt. Um die Nähe (Abstände) dieser Punkte zueinander zu berechnen, wird die euklidische Distanz verwendet. Sowohl unter der einfachen Bedingung ($t(17.17) = -19.39, p < 0.01$) als auch unter komplexeren Bedingung ($t(21.89) = -20.97, p < 0.01$), wies die EMME-Gruppe signifikant kleinere euklidische Distanzen im Vergleich zur Kontrollgruppe auf.

Tabelle 6.11: Euklidische Distanzen (Mittelwert (Standardabweichung)) für EMME-Gruppe (EG) und Kontrollgruppe (KG)

Bedingung	EG	KG
Einfach	170.44px (14.25)	334.95px (25.74)
Komplex	201.43px (23.68)	397.36px (22.06)

Der signifikante Unterschied, d. h. kleinere Mittelwerte der Euklidischen Distanzen um ca. 160 Pixel im einfachen und 180 Pixeln im schwierigen Lernvideo (siehe Tabelle 6.11), zeigt, dass Teilnehmer der EMME-Gruppe ihren Blick und damit die Aufmerksamkeit von Blicken des Experten leiten lassen.

Durch das EMME scheint es daher zu gelingen, den Blick der Teilnehmer der EG bzw. die Aufmerksamkeit der Teilnehmer zu lenken. Dieser Wert ist besonders als Voraussetzung für weitere Auswertungen zu sehen, denn wenn die euklidische Distanz der Kontrollgruppe geringer gewesen wäre, hätte dies bedeutet, dass die Fokuslenkung nicht funktioniert hätte und alle anderen Analysen nicht sinnvoll wären.

6.11.3.2 *Erkenntnisse zur zweiten Hypothese*

Ziel der zweiten Hypothese ist die Überprüfung, wie das EMME von Studierenden bewertet wird und Indikatoren dafür abzuleiten, ob ein EMME als Scaffold geeignet wäre. Dazu wurde der Fragebogen bzw. die Items, die in Abschnitt 6.11.2.7 beschrieben sind, nach jeder Lernphase gestellt. Für die Bewertung wurden Item 1, 2 und 3 herangezogen.

Tabelle 6.12: Ergebnisse (Mittelwerte (Standardabweichung)) der Bewertung

Gruppe	Item 1	Item 2	Item 3
EG einfache Bedingung	2.91 (0.94)	3.36 (0.81)	2.45 (0.69)
KG einfache Bedingung	3 (0.85)	3.42 (0.9)	2.08 (0.79)
EG komplexe Bedingung	3 (1)	3.27 (1.01)	3.27 (0.79)
KG komplexe Bedingung	2.42 (1.08)	2.83 (0.83)	2.92 (1.08)

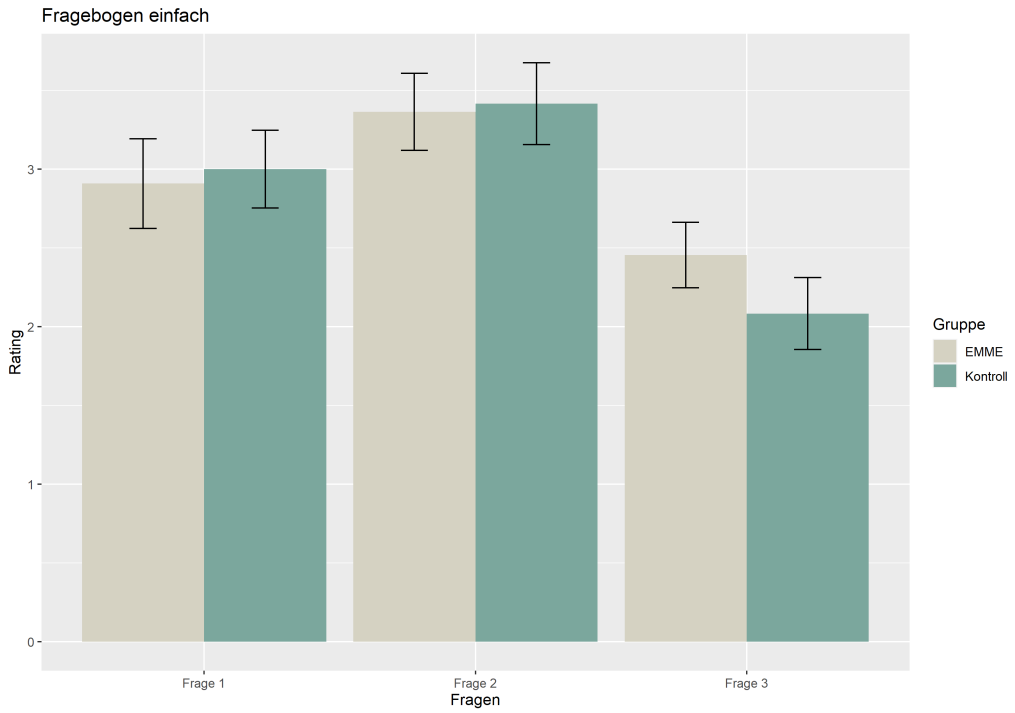


Abbildung 6.52: Ergebnisdarstellung im Vergleich der beiden Gruppen für die Evaluation des Lehrvideos mit den einfachen Inhalten

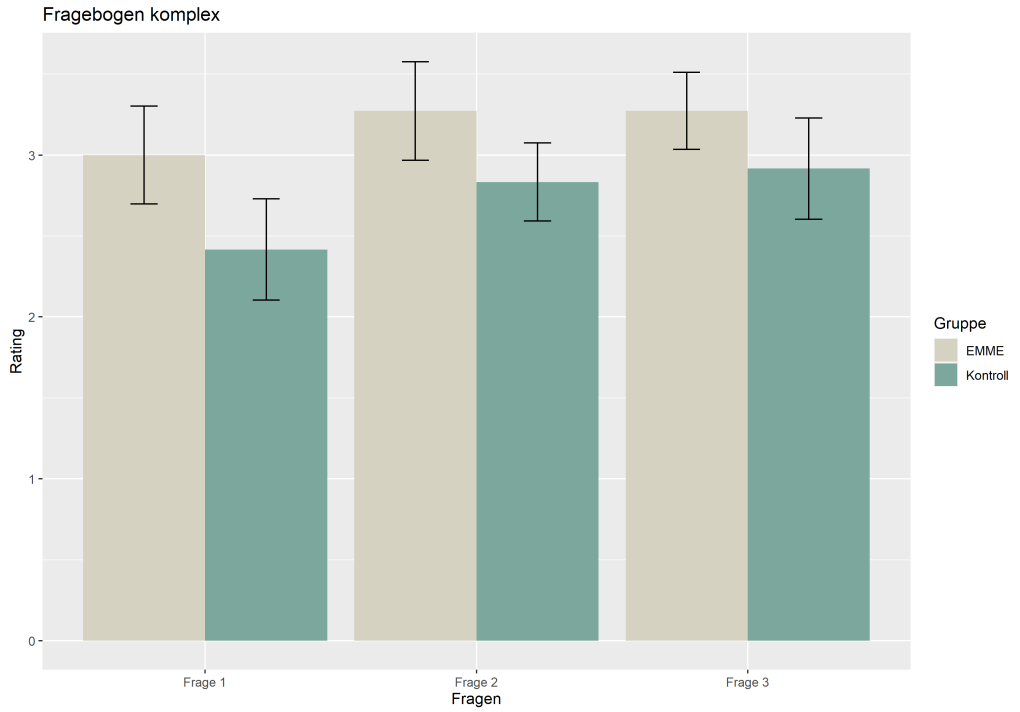


Abbildung 6.53: Ergebnisdarstellung im Vergleich der beiden Gruppen für die Evaluation des Lehrvideos mit den komplexen Inhalten

Bei den Ergebnissen zeigen sich für beide Schwierigkeitsgrade (einfache und komplexe Bedingung) bei keinem der drei Items signifikante Unterschiede zwischen den beiden Gruppen (für alle Items $p > 0.05$). Die Ergebnisse sind tabellarisch in Tabelle 6.12 dargestellt und visualisiert in Abb. 6.52 und Abb. 6.53. Die gestellte Hypothese kann deshalb nicht bestätigt werden. Es lässt sich lediglich ableiten, dass von den Teilnehmern keine Videoform als besser oder schlechter bzw. unterstützender oder hinderlicher bewertet wird. Abbildung 6.52, in der Bewertungen abgebildet sind, die das Video zur einfachen Bedingung evaluieren, zeigt keine Tendenz. Abbildung 6.53, in der Bewertungen abgebildet sind, die das Video evaluieren, das vor der Bearbeitung der komplexeren Aufgabe evaluieren zeigt eine eher positive Tendenz in der Bewertung der EMME-Gruppe.

Zusatz:

Das dritte Item mit der Frage nach dem Schwierigkeitsgrad des Diagramms wird im Sinne eines „Qualitätskriteriums“ geprüft, ob die „empfundene“ Schwierigkeit, der Einschätzung der Forscher entspricht. Über beide Gruppen hinweg (also EMME- und Kontrollgruppe kombiniert) zeigt sich, dass das einfache EMME als „einfacher“ bewertet wird $(2.26(0.75))$, und das EMME mit dem komplexeren Inhalt auch als schwie-

riger bewertet wird (3.09(0.95)). Dieser Unterschied erreicht statistische Signifikanz[36].

6.11.3.3 *Erkenntnisse zur dritten Hypothese*

Die dritte Hypothese formuliert bezogen auf die CLT beschrieben in Abschnitt 3.2.1 eine größere Steigerung des Cognitive Loads (und zwar des gesamten Load, da hier nicht differenziert werden kann) bei der Kontrollgruppe, da angenommen wird, dass das EMME als Scaffold dient, das die Teilnehmer kognitiv entlastet (van Gog, Jarodzka, Scheiter, Gerjets & Paas, 2009; Sweller, 1988; Sweller et al., 1998). Wie in Abschnitt 3.4 und Abschnitt 3.2.1 beschrieben, sollten Unterrichtsmaterialien so gestaltet sein, dass kognitive Ressourcen optimal genutzt werden. Wie oben bereits beschrieben, zählen EMMEs zu den *worked examples* und es wurde bereits gezeigt, dass sich diese für Anfänger eignen (z. B. Atkinson, Derry, Renkl & Wortham, 2000; Sweller et al., 1998), (Jarodzka, Van Gog, Dorr, Scheiter & Gerjets, 2013). Es wird angenommen, dass eine stärkere Erweiterung der Pupillen, also ein steigender Pupillendurchmesser mit steigender Belastung einhergeht (Holmqvist et al., 2013, S. 393).

Für die Durchführung der Studie wurde diese Annahme wie oben beschrieben operationalisiert und ein inhaltsleerer Baseline-Stimulus für jede Gruppe generiert, für 5 Sekunden gezeigt und dabei der Pupillendurchmesser gemessen und als „Grundbelastung" festgelegt (siehe Holmqvist et al., 2013, S. 393). Konkret wurden die Baseline-Werte für die letzten beiden Sekunden des fünfsekündigen Stimulus für die Grundbelastung herangezogen, da man davon ausgeht, dass in diesem Zeitraum die Pupillen eines Teilnehmers bereits an die Lichtverhältnisse angepasst sind. Zudem wurde der Durchmesser der Pupille (über beide Augen gemittelt) während der ersten Lernphase ermittelt.

36 Ein Shapiro-Wilk-Test auf Normalverteilung ergab, dass weder die Antworten auf das Item Schwierigkeit ($W = .79, p < .01$) noch auf das Item Schwierigkeit (W = .91, p = .04) eine Normalverteilung aufweisen. Daher wurde ein Wilcoxon-Test (Wilcox, 2012) für gepaarte Stichproben (bzw. Wilcoxon signed rank Test mit Kontinuitätskorrektur) berechnet. Es zeigte sich ein signifikanter Unterschied ($V = 0, p < .01$) zwischen der Bewertung des Items Schwierigkeit, $M = 2.26 (SD = 0.75)$ und des Items Schwierigkeit, $M = 3.09 (SD = 0.95)$.

Abbildung 6.54: Erweiterung der Pupillendurchmesser über die Zeit des Lehrvideos der einfachen Lernphase

Tabelle 6.13: Gruppierte Werte für Pupillendurchmesser: Baselinewert, berechnet für die letzten beiden Sekunden (Sekunde 4 und 5) des Baseline-Stimulus; Baselinedifferenz, Veränderung des Pupillendurchmessers in Millimeter; Dabei wird der Baseline-Wert vom Pupillendurchmesser während der Lernphase subtrahiert; Manipulierte Baseline, Sekunde 5-10 während der Lernphase 1 werden zur Berechnung des Baseline Stimulus verwendet; Differenz zur manipulierten Baseline,Veränderung des Pupillendurchmessers in Millimeter, dabei wird der Baseline-Wert vom Pupillendurchmesser während der Lernphase subtrahiert.

Gruppe	Baseline	Baseline-differenz	Manipulierte Baseline	Differenz zur manipulierten Baseline
EG	3.84 (0.61)	-0.47mm (0.2)	3.49mm (0.58)	-0.12mm (0.11)
KG	3 (0.35)	-0.09mm (0.13)	3.23mm (0.37)	-0.32mm (0.15)

Es gibt einen signifikanten Unterschied in der Pupillenerweiterung zwischen den beiden Versuchsgruppen. Die EMME Gruppe ($N = 11$) zeigte eine signifikant stärkere Abweichung vom Baseline Wert ($Mittelwert(M) = -0.47, (Standardabweichung(SD) = 0.2)$) als die Kontrollgruppe
($N = 10; M = -0.09 (SD = 0.13); t(17.3) = -5.11, p < .01$). Ein Teilnehmer der EMME-Gruppe führte das Experiment nicht in der gleichen Umgebung durch, wie alle anderen Teilnehmer und wurde wegen unterschiedlicher Lichtverhältnisse ausgeschlossen. Zusätzlich sind 2 Teilnehmer aus der Kontrollgruppe ausgeschlossen, da ihnen der Basisreiz nur für eine Sekunde und nicht wie vorgesehen für 5 Sekunden präsentiert wurde. Dies ist auf eine falsche Einstellung in der Eye-Tracking-Software bei den ersten beiden Teilnehmern der Kontrollgruppe zurückzuführen, wurde aber für die folgenden Teilnehmer korrigiert.

Die Grafik (Abb. 6.54) veranschaulicht die gruppierten Veränderungen des Pupillendurchmessers über die gesamte Zeit des ersten Lernvideos hinweg. Die jeweils helleren Linien stellen die gemittelten gruppierten Pupillenveränderungen über die Zeit hinweg dar. Ausreiser sind so zu erklären, dass möglicherweise für eine Sekunde (bzw. Mikrosekunde) nur ein (oder wenige Werte) Wert (für einen Teilnehmer) vorhanden ist und dieser so stark ausreißt. Die jeweils dunkleren Linien in der Grafik stellen eine Trendlinie dar und sind eine vereinfachte Variante

der helleren Linien. Ausreiser sind so zu erklären, dass möglicherweise für eine Sekunde (bzw. Mikrosekunde) nur ein (oder wenige Werte) Wert (für einen Teilnehmer) vorhanden ist und dieser so stark ausreißt. Für die vereinfachte Darstellung wurde jede 300ste Wert verwendet, Daten dazwischen wurden linear interpoliert. Im Bezug auf den Baseline-Stimulus (Wert 0 der x-Achse) haben sich die Werte beider Gruppen tendenziell verringert, weswegen im gesamten Zeitverlauf hinweg auch negative Werte zu verzeichnen sind.

Bezogen auf die Grafik sind die Ergebnisse so zu interpretieren, dass größere negative Abweichungen größere Verkleinerungen der Pupille bedeuten und somit mit geringerem Cognitive Load assoziiert werden können (siehe Holmqvist et al. (2013), 393). Da die EMME-Gruppe eine größere negative Abweichung von der Baseline verzeichnet, bedeutet dies eine stärkere Verkleinerung der Pupille. Somit kann man interpretieren, dass die EMME-Gruppe während des Betrachtens des Videos einen geringeren kognitiven Load aufwies als die Kontrollgruppe, was zu einer Bestätigung der Hypothese führt. Allerdings sind hier folgende Einflüsse in Bezug auf die Interpretation der Daten zu beachten:

- Die negativen Werte über den Zeitverlauf hinweg deuten darauf hin, dass die kontrollierte Experimentalumgebung hier an ihre Grenzen stößt. Es ist möglich, dass der Baseline-Stimulus nicht geeignet war. Der nachfolgende Lichtkegel des EMME-Videos (Abb. 6.50) hatte möglicherweise einen zu großen Kontrast zum Baseline Stimulus.
- Außerdem konnte keine konstante Leuchtkraft (constant luminocity) gewährleistet werden (der Raum, in dem die Daten erhoben wurden, war abgedunkelt und mit chemischem Licht ausgestattet, die Abdunkelung konnte aber nicht vollständig gewährleistet werden). Es wurde ein sogenannter dynamischen Stimulus verwendet (das Video), wodurch die Kontrolle der Leuchtkraft eher schwierig ist. Laut Holmqvist et al. (2013) ist besonders bei der Interpretation von Pupillendaten in Videos Vorsicht geboten.

6.11.3.4 *Erkenntnisse zur vierten Hypothese*

Die folgenden beiden Hypothesen 4 und 5 beschäftigen sich mit der Rolle der Performanz der Teilnehmer.

Es wird erwartet, dass die Teilnehmer der EMME-Gruppe im Posttest eine bessere Leistung zeigen, als Teilnehmer der Kontrollgruppe. Dazu wird die Leistung anhand zweier Tests (einfach und komplex) gemessen und bewertet (siehe Abschnitt 6.11.2.6). Ergebnis der Bewertung ist eine Zahl zwischen 0 und 78, welche die Qualität des erstellten Diagramms abbildet und somit als Performanz-Maß des Teilnehmers gilt.

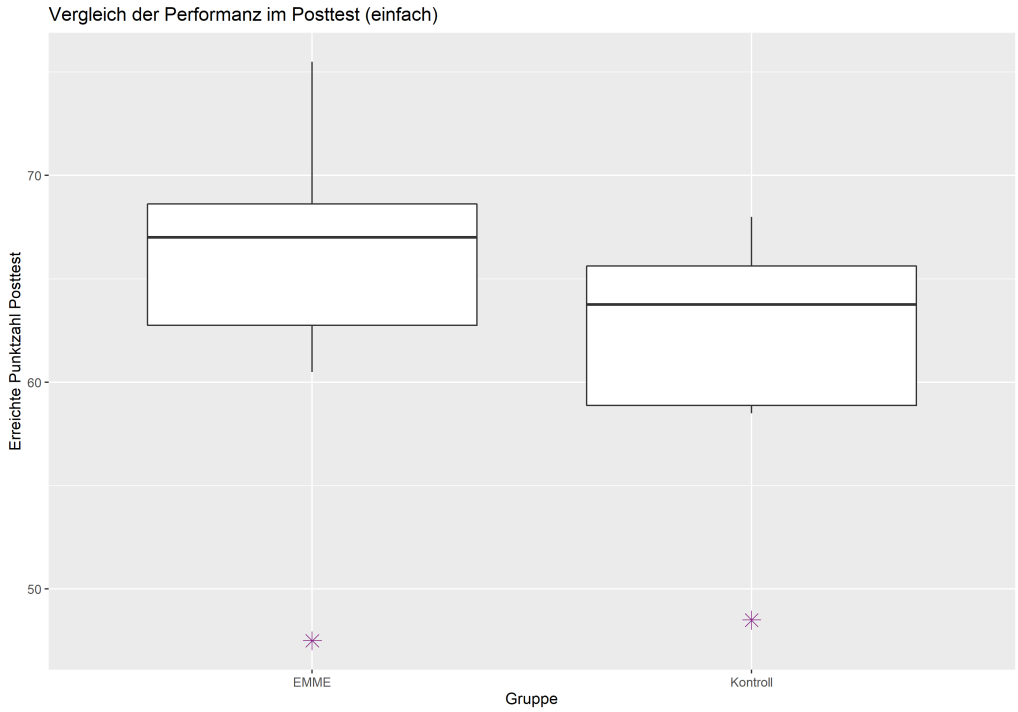


Abbildung 6.55: Leistung EMME-Gruppe und Kontrollgruppe für den einfachen Posttest

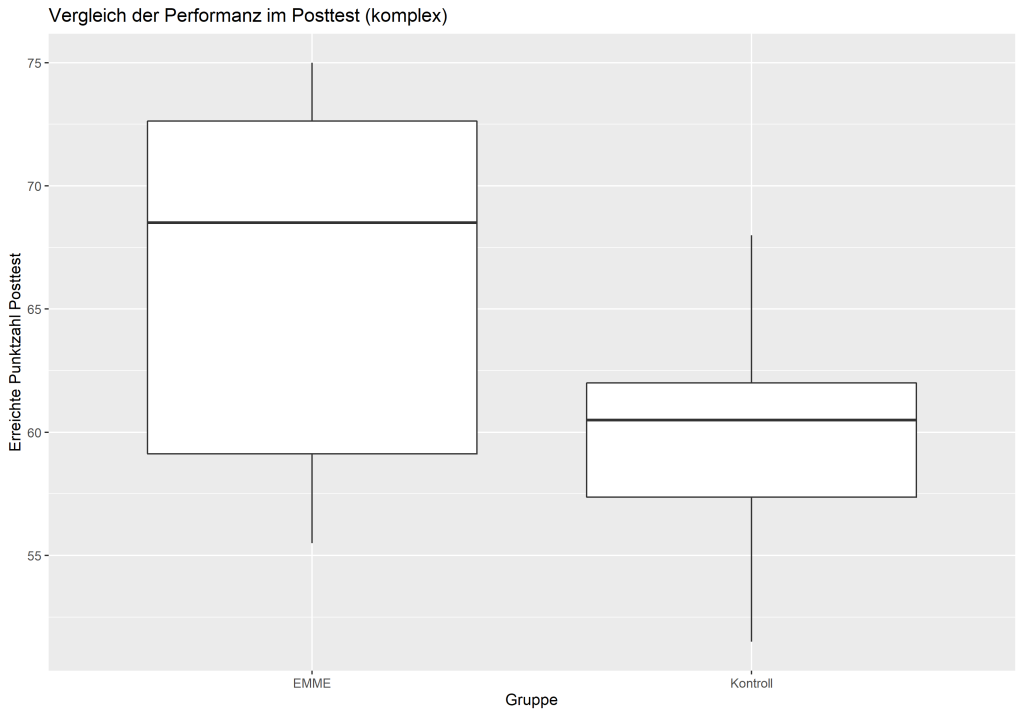


Abbildung 6.56: Leistung EMME-Gruppe und Kontrollgruppe für den komplexen Posttest

Tabelle 6.14: Erzielte mittlere Punkte im Vorwissenstest und den beiden Posttests für die EMME-Gruppe (EG) und die Kontrollgruppe (KG)

Gruppe	Vorwissenstest	Posttest (einfach)	Posttest (komplex)
EG	7.04 (2.68)	65.33 (6.89)	66.33 (7.44)
KG	7.29 (2.72)	62.17 (5.54)	59.58 (4.59)

Insgesamt zeigen die Ergebnisse in Abb. 6.55 und Abb. 6.56, dass die Teilnehmer der EMME-Gruppe einen höheren Mittelwert bei der Performanz-Bewertung $M = 65.33(SD = 6.89)$ als die Kontrollgruppe $M = 62.17(SD = 5.54)$ erreichen. Für den Posttest, der nach der ersten Lernphase stattfand und einfacher gestaltet war, wurden keine signifikanten Unterschiede erzielt ($t(21.042) = 1.2407, p = 0.23$). Für den zweiten, komplexeren Posttest, erreichten die Teilnehmer der EMME-Gruppe ebenfalls einen höheren Mittelwert bei der Performanz-Bewertung $M = 66.33(SD = 7.44)$, als die Kontrollgruppe $M = 59.58(SD = 4.59)$, wobei hier der Unterschied zwischen den beiden Gruppen auch Signifikanz erreicht ($t(18.324) = 2.6757$, $p = 0.015$). Teilnehmer der EMME-Gruppe zeigen demnach einen bessere Leistung, auch wenn der Unterschied nur für den Posttest der zweiten Lernphase signifikante Ergebnisse zeigt. Die Hypothese, dass sich EMMEs positiv auf die Leistung abbilden, kann bestätigt werden.

6.11.3.5 *Erkenntnisse zur fünften Hypothese*

Final soll nun noch beschrieben werden, ob das Vorwissen der Teilnehmer die Performanz der Teilnehmer im Posttest in den beiden Schwierigkeitsstufen beeinflusst. Ergebnis des Vorwissenstests ist eine Zahl zwischen 0 und 15, die das Vorwissen des Teilnehmers abbildet. Die erfasste Performanz zum Vorwissen (siehe Anhang A.6 und Tabelle 6.14) wurde mit den Ergebnissen der Posttests des jeweiligen Teilnehmers gruppiert und miteinander in Beziehung gesetzt.

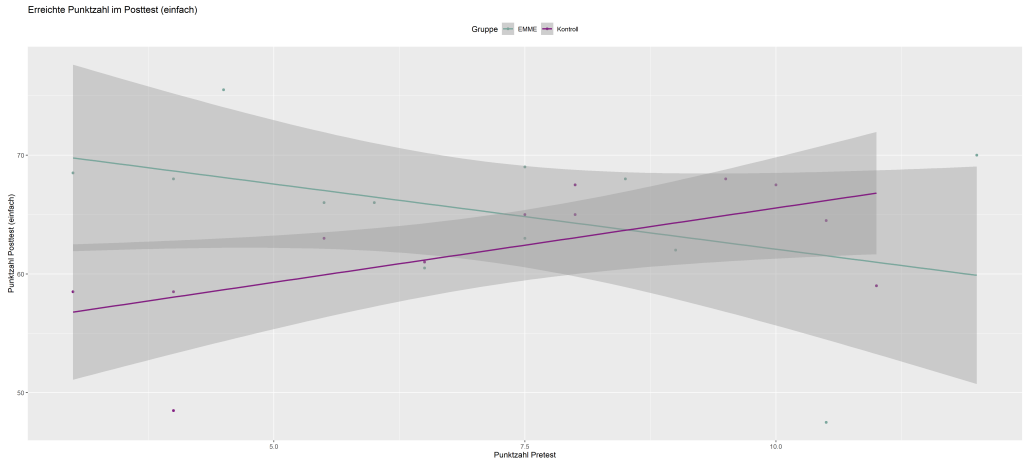


Abbildung 6.57: Korrelation der Pretests mit den Posttest für das einfache Szenario

Abbildung 6.57 zeigt, dass die Teilnehmer der EMME-Gruppe, die im Vorwissenstest schlechter abschneiden, im Posttest besser abschneiden, als Teilnehmer, die im Vorwissenstest bereits gut abschneiden. Dies kann so interpretiert werden, dass das EMME größeren Nutzen für Personen zeigt, die im Pretest tendenziell schlechter abschneiden, als bei Personen, die im Pretest bereits besser abschneiden. Die Kontrollgruppe hingegen zeigt erwartetes Verhalten, indem bereits im Pretest als besser identifizierte Personen auch im Posttest bessere Diagramme erstellen.

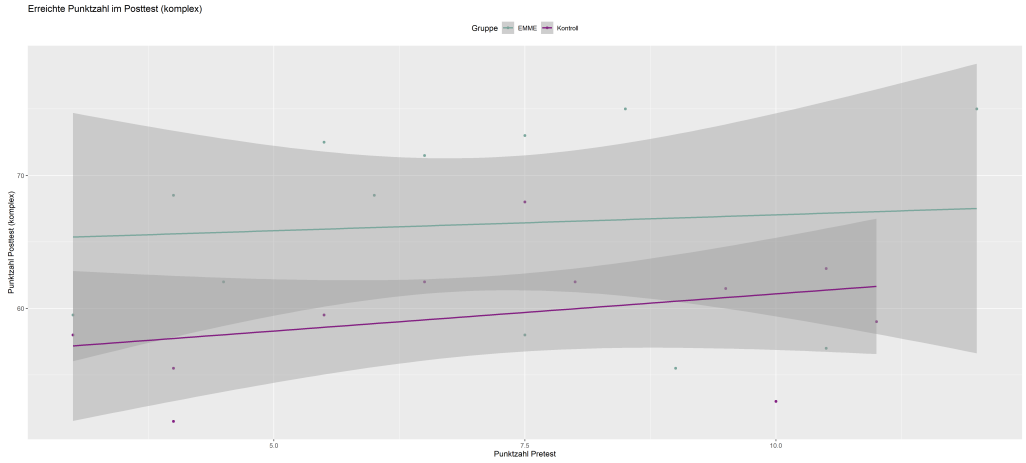


Abbildung 6.58: Korrelation der Pretests mit den Posttest für das komplexe Szenario

Abb. 6.58 zeigt, dass das EMME gleichermaßen einen Nutzen sowohl für stärkere als auch für schwächere Personen zeigt (bewertet anhand des Pretests). Dies ist dadurch zu erklären, dass die schwierige Aufgabe auch für die stärkeren Personen fordernd ist, was auch durch das dritte

Item des Fragebogens (siehe Hypothese zwei) bestätigt wurde. Möglicherweise war in der ersten Phase des Experiments die einfache Aufgabe für stärkere Personen nicht ausreichend fordernd. Man könnte vermuten, dass das EMME deshalb nicht so nützlich war wie für schwächere Teilnehmer. Schlussfolgernd lässt sich festhalten, dass Personen kognitiv gefordert sein müssen, damit EMME Videos Nutzen im Hinblick auf die Performanz entfalten.

6.11.4 *Interpretation der Daten*

Insgesamt lässt sich über die Ergebnisse der Auswertung folgendes Fazit ziehen:

- Die Studie weist Indizien dafür auf, die dafür sprechen, dass EMMEs im Gegensatz zu „konventionellen" Videos eine geringere kognitive Belastung bei Lernern hervorrufen. Dies muss allerdings unter kontrollierten Bedingungen und einem größeren Sample noch gezeigt werden.
- EMMEs können positiven Effekt auf die Performanz eines Lerners haben.
- Damit EMMEs ihre Wirkung zeigen, muss der Lerner kognitiv gefordert sein, da schwächeren Personen nachdem sie das EMME gesehen haben, besser abschneiden während dies bei insgesamt wissensstärkeren Lernen nicht der Fall ist. Dies wurde bereits in vorherigen Studien gezeigt (siehe Mason, Pluchino und Tornatora (2016)).
- Das EMME wurde im Vergleich zu dem „konventionellen" Video nicht schlechter, aber auch nicht positiver von den Teilnehmern des Experiments bewertet. Über die Eignung als Scaffolds aus Sicht der Lerner kann keine Aussage getroffen werden.

6.12 Pädagogisch-didaktische Verortung & Rückbezug

Die Modellierung eines Softwaresystems mit der UML gilt als komplexer Themenbereich im Software Engineering, in welchem nicht immer ein klar definierter Lösungsweg existiert, sondern es vor allem verschiedene korrekte Lösungen geben kann. Zur Unterstützung in diesem Problemlösevorgang können Hilfsmittel bzw. Unterstützungsmaßnahmen angeboten werden, die in verschiedenen Theorien, die sich mit Instruktionsdesign beschäftigen, als Scaffolds bezeichnet werden. Instruktionsdesign ist dabei zu verstehen als systematischer Ansatz zur Konzeption von Lernumgebungen und den zugehörigen Materialien.

Scaffolds finden ihren Einsatz im Konzept des Scaffolding. Dieses stellt ein grundlegendes Konzept im Rahmen der Methodik des „Cognitive Apprenticeship“ (van Merriënboer, Kirschner & Kester, 2003) dar: Scaffolding bezieht sich auf eine Kombination aus Unterstützung für die Performanz und „Fading“ zum Erreichen eines Lernziels. Dabei ist es wichtig, passende Art und Intensität an Unterstützung zu finden und sie anschließend angemessen zurückzunehmen (fading) (van Merriënboer et al., 2003, S.4).

Spezifisch handelt es sich in diesem Vorhaben um sogenannte „Hard Scaffolds“: dies sind statische Unterstützungen, die im Vornhinein antizipiert und geplant werden können, basierend auf Schwierigkeiten, die typischerweise bei Studierenden auftreten. Beispielsweise kann man konzeptuelle Hilfsmittel bereitstellen, die Studierenden dabei helfen, Wissen zu organisieren und verbinden. Alle vorgestellten Scaffolds können dieser Kategorie zugeordnet werden. Im Gegensatz dazu sind sogenannte „Soft Scaffolds“ eine Form von Hilfsmitteln, die dynamisch und situationsgebunden sind. Diese Art Scaffolds benötigen eine Lehrperson die kontinuierlich analysiert, was die Lernenden verstehen und basierend darauf Unterstützung anbietet (Saye & Brush, 2002).

Während Studierende eine Aufgabe bearbeiten, sollen sie jederzeit und unabhängig von Fortschritt oder Schwierigkeitsgrad auf die geplanten Hilfsmittel zugreifen können. Somit können sie zu jedem beliebigen Zeitpunkt, während des Arbeitens an einer Aufgabe unterstützt werden. Dadurch, dass die zu bearbeitenden Lernaufgaben zwar konstruiert sind aber stets authentischen Charakter aufweisen, können Studierende schrittweise zur Unabhängigkeit von Hilfsmitteln herangeführt werden und erlernte Skills integrieren van Merriënboer et al. (2003). Durch die Bereitstellung von Scaffolds kann deren kognitive und metakognitive Entwicklung unterstützt werden.

Die Bereitstellung von Hilfsmitteln soll in einer konstruktivistischen Lernumgebung geschehen. Mit ebendiesem Design konstruktivistischer

Lernumgebungen für das Erlernen von Problemlösen und konzeptueller Entwicklung in Problembereichen, welche als *ill-defined* bzw. *ill-structured* (schwach definiert bzw. schwach strukturiert) gelten, befasst sich eine Theorie von Jonassen (1997). *Ill-structured* Probleme sind gekennzeichnet durch unbestimmte Zielzustände, mehrere Lösungen oder Lösungswege und weisen Unbestimmtheiten bezüglich Konzepten, Regeln und Prinzipien auf. *Ill-structured* Probleme sind kontextabhängig und es ist nötig, eine authentische Aufgabenumgebung zu entwickeln (Jonassen, 1997). Dies trifft auf Software Engineering und auch auf die hier fokussierte Subdisziplin der Modellierung zu.

Eine konstruktivistische Lernumgebung Abschnitt 3.3 ist nach Jonassen (1997) wie folgt aufgebaut: Der Lernende kann das Lernziel selbst festlegen, das Lernen ist authentisch und aktiv und die Lernumgebung beinhaltet Erfahrungen, die die Wissenskonstruktion begünstigen. Zudem sollten dem Lernenden *just in time*-Informationen zur Selbstauswahl, demnach Hilfsmittel, bereitstehen. Kognitive Werkzeuge zur Repräsentation von Problemen, zur Darstellung von Wissen, zur Unterstützung der Performanz und im Erhalt von Informationen sollen dem Lernenden gescaffolded (d. h. stufenweise) angeboten werden. Soziale bzw. kontextuelle Unterstützung soll außerdem in der Lernumgebung dargeboten werden, indem der Lernende beispielsweise durch Anpassen der Aufgabenschwierigkeit gescaffolded (stufenweise unterstützt) wird. Grundlegend zur Anwendung dieser Theorie ist, dass ein zugrundeliegendes Problem Antrieb des Lernens ist. Die Studierenden erlernen domänenspezifisches Wissen um ein Problem zu lösen (im Gegensatz zum Problemlösen als Wissensanwendung) (Jonassen, 1997).

6.13 STUDIEN ZUM EINSATZ VON SCAFFOLDS IN DER SOFTWARE ENGINEERING-LEHRE

Trotz verbreiteter Forschung zu Lernhilfen, gestuften Lernhilfen und dem sogenannten Scaffolding in vielen Bereichen der Schulforschung, vor allem im Bereich der Naturwissenschaften, findet sich für die Hochschullehre und den Bereich Software Engineering noch kaum Literatur zu schematischen Vorgehensweisen zum Aufbau von Hilfestellungen. Vielmehr lassen sich gerade auf Konferenzen, auch im Bereich der Software Engineering- Lehre, eine Vielzahl einzelner Untersuchungen mit den verschiedensten Materialien und Herangehensweisen identifizieren, die aber nicht unbedingt den Aufbau einer Hilfestellung oder Arten von Hilfestellungen in den Vordergrund stellen.

Eine Literaturrecherche nach dem bereits vorgestellten Prinzip in Abschnitt 5.2 wurde durchgeführt, um bereits existierende Hilfsmittel bzw.

Lernhilfen für die Software Engineering- Lehre zu identifizieren. Durchsucht wurden ebenfalls die bereits in Abschnitt 5.2.1 genannten Datenbanken, wobei folgender Suchterm, abgeleitet aus der Forschungsfrage FF2.1 (Brereton et al., 2007, 4), verwendet wurde:

((Scaffolding OR scaffolds) AND (teaching OR learning OR education OR student) AND „software engineering“))

Sofern vorhanden, wurde für die einzelnen Suchanfragen ein erweiterter Suchmodus genutzt, um den Suchterm nutzen zu können. Der Suchterm musste an die Abfragesprache der jeweiligen Datenbank angepasst werden. Dabei wurde lediglich die Syntax des Terms verändert, damit eine gemeinsame Betrachtung in der späteren Analyse ohne Einschränkungen durchgeführt werden kann.

Zur Filterung nach Titel und Abstract der Ergebnisse und zur finalen Auswahl in Stufe 2 wurden folgende Inklusions- und Exklusionskriterien gewählt:

INKLUSIONSKRITERIEN

- Das Literaturformat ist entweder ein Buch (Monographie), ein Journal- oder Magazinbeitrag, ein Beitrag in einem Tagungsband, Konferenz- band oder Sammelwerk, eine Dissertation oder ein Report.
- Das Material ist in Englisch oder Deutsch verfasst.
- Das Material enthält (in Titel und/oder Abstract) Textsegmente die darauf hindeuten, dass in dem Material **Scaffold**, **Scaffolding**, **Hilfestellung** oder **(gestufte) Lernhilfen**, **Unterstützungsmöglichkeiten** oder synonyme Begriffe zu Lernhilfen behandelt werden.
- Das Material identifiziert konkrete Hilfsmittel, Hilfestellungen, Unterstützungsmöglichkeiten zum Thema Software Engineering (hierzu zählt auch die Programmierausbildung) oder fachunabhängig in der **Hochschullehre**.

EXKLUSIONSKRITERIEN

- Das Material enthält keine Textsegmente zu „Scaffold“, „Scaffolding“ oder „(gestufte) Lernhilfen“.
- Das Material beschreibt keine konkreten Hilfestellungen zum Thema Software Engineering oder Informatik.
- Das Material beschreibt keine konkreten Hilfestellungen in der Hochschullehre.
- Das Material beschreibt ein Lehrexperiment ohne Fokus auf den Aufbau der Lernhilfe bzw. des Hilfsmittels.

Tabelle 6.15: Durchsuchte Datenbanken und die Summe der identifizierten Publikationen in Spalte 1 sowie die Anzahl der identifizierten Veröffentlichungen nach Filterung nach Titel und Abstract in Spalte 2

Datenbank	# Gefundene Paper	# Nach Titel & Abstract	# Relevanz
ACM	17	4	2
ERIC	3	1	0
IEEE	26	11	6
Sagejournals	39	0	0
Scopus	60	17	5
Wiley	6	4	1
Sciencedirect	224	18	6
Web of Knowledge	5	1	1
Springer	0	0	0
	380	56	21

Zunächst wurde das Material wie in Abschnitt 5.2 nach Titel und Abstract gefiltert (Tabelle 6.15, Spalte 3). Mit diesen Ergebnissen wurde Stufe 2 des Prozesses gestartet. Duplikate wurden entfernt und Experten wurden nach weiterer Literatur befragt. Eine weitere tabellarische Darstellung der Publikationen erfolgt nicht, da Publikationen in mehreren Datenbanken dupliziert vorkamen verschwimmt die Zuordnung der Publikation zur Datenbank. Es ergaben sich *N=56* Publikationen zur weiteren Analyse. Diese wurden dann codiert, Inklusionskriterien und Exklusionskriterien angewendet und *N=21* Publikationen identifiziert, die Scaffolds rund um das Thema Software Engineering für alle Phasen des Software Lebenszyklus beschreiben. Die Intention der Arbeit ist allerdings, Scaffolds zu identifizieren, die Unterstützung im Softwaredesign oder der Softwarearchitektur mit der UML bieten. Deshalb werden im Folgenden die *N=21* identifizierten Publikationen noch einmal gefiltert, um konkret Scaffolds für die Modellierung, Softwaredesign und Softwarearchitektur zu identifizieren. In diesem Prozess wurden *N=5* Publikationen identifiziert. Um auch die Schwerpunkte der Betrachtung von Scaffolds in der existierenden Literaturlandschaft zu verdeutlichen, wurden für die übrigen Publikationen Kategorien gebildet: Programmierung, Softwareentwicklungsprozess, Datenbanken, Projektmanagement und Requirements Engineering (Requirement Engineering). Mit *N=8* Publikationen ist der Bereich der Programmierausbildung am stärksten untersucht, gefolgt von Unterstützungsmöglichkeiten für den Software-

entwicklungsprozess im Allgemeinen oder für alle Phasen des Prozesses und Projektmanagement mit je *N=3* Publikationen. Für die übrigen Kategorien Datenbanken und Requirements Engineering wurde jeweils nur eine Publikation identifiziert.

Im Folgenden werden zunächst die für die Fragestellung relevanten Publikationen kurz beschrieben, anschließend Diejenigen, die für die einzelnen Bereiche Identifizierten (Anmerkung: Die Scaffolds sind fett markiert.).

6.13.1 *Identifizierte Publikationen: Scaffolds für UML, Softwaredesign, Softwarearchitektur*

Feldgen und Clua (2012) betrachten in ihrer Veröffentlichung den Entwurf verteilter Systeme und die Schwierigkeit für Novizen diesen zu realisieren. Sie vertreten die Thesen, dass Novizen dazu neigen, linear zu denken und sich nur auf das vorliegende Problem zu konzentrieren. Zudem verwenden sie häufig Trial-Error-Strategien und es fehlt ihnen an Vertrauen in ihre Entwurfsentscheidungen. Diesen Problemen versuchen sie mit Hilfe von **Modellierungsstrategien von Experten**, **Peer Review** und sogenannten **Question Prompts** zu begegnen. Letztere sind entweder Erläuterungen oder metakognitiven Fragen um einzelne Problemlösungsschritte zu unterstützen.

Reddy, Iyer und Sasikumar (2017) stellen eine Technology Enhanced Learning (TEL)- Umgebung vor, die Unterstützung zur Entwicklung von divergierendem und konvergierendem Denken bieten soll. Divergentes Denken beinhaltet laut Reddy et al. (2017) Prozesse, wie das Verstehen des Problems aus mehreren Perspektiven und das Generieren mehrerer Lösungen für ein Problem. Konvergentes Denken hingegen ist die Bewertung und Auswahl der Lösung auf der Grundlage der Kriterien und Einschränkungen. Nach Reddy et al. (2017) sind diese Fähigkeiten im Softwareentwicklungsprozess wichtig um bessere Entwurfsergebnisse zu erreichen. Die Autoren bieten **Eingabeaufforderungen (Prompts)**, **Beispiele**, **Simulationen** und **spezielle Tools** an. Die divergierenden Denkaktivitäten werden mit einem Zeichenwerkzeug unterstützt, um die mentalen Modelle des Systems zu modellieren und mit einer Attributlistenkarte eine Lösung zu generieren. Die Aktivitäten des konvergenten Denkens werden mit Pro- und Contra-Analysetabellen, Bewertungstabellen und Entscheidungstabellen unterstützt, die mit Lösungen und Kriterien ausgestattet sind, die von den Studierenden generiert wurden, um den Ablauf von einem Schritt zum Anderen aufrechtzuerhalten. Alle Aktivitäten werden je nach Bedarf unterstützt mit domänenspezifischen

Hinweisen in Form von **Eingabeaufforderungen**, **Erläuterungen** und **Simulationen**.

Manno et al. (2011) betrachten in ihrer Arbeit das kollaborative Arbeiten Studierender während der Modellierung eines Use-Case-Diagramms. Dabei wurde eine **Eigenentwicklung der Autoren, das Shared Drawing Tool**, sowie ein **Chattool** eingesetzt. Das Experiment war als Kontrollgruppenexperiment angelegt, wobei eine Hälfte der Gruppe zunächst Face-to-Face modellierte, die andere Hälfte computergestützt mit Hilfe des Tools. Anschließend wurde getauscht. Die Forscher fanden heraus, dass die Gruppen, wenn sie in computergestützter Umgebung modellierten besser abschnitten, als ohne diese.

Im Fokus von Rosson und Carroll (1996) steht objektorientiertes Design. Die Autoren betrachten sogenannte **scaffolded examples**. Diese Beispielprobleme haben eine realistische Größe und deren Komplexität wird nicht gleich (und damit möglicherweise überfordernd) zu Beginn, sondern Schritt für Schritt aufgezeigt. Die Beispielprobleme beginnen mit Nutzer-Interaktionsszenarien, die je einen konkreten Teil eines Design Problems beinhalten und eine Beschreibung eines Nutzers der ein bestimmtes Ziel damit verfolgt. Laut Rosson und Carroll (1996) helfen diese Szenarien dabei ein großes, abstraktes Problem in ein kleine, konkreten Unterprobleme zu zerlegen. Damit geht der Fokus auf die Objektinteraktionen nicht verloren und es wird eine einfache Methode zur Identifikation von notwendigen Objekten eingeführt: Beginne mit den Entitäten, die in der Beschreibung des Nutzers erwähnt werden. Zusätzlich zu dem Szenario gibt es ein **selbst entwickeltes Tool der Autoren, um ein Szenario aus der Perspektive eines Design Objekts analysieren** zu lassen. Jedes identifizierte Objekt der Nutzerbeschreibung hat sein eigenes Szenario erhalten. Durch diese Perspektive wird das OOD Konzept der Kapselung und Kollaboration hervorgehoben. Mit Hilfe dieses Werkzeugs kann die weitere Basis gelegt werden für einen Objektinteraktionsgraphen.

Thomasson et al. (2006a) stellt das Tool **VorteX** (Visually Oriented Training Experience) vor, das Studierenden statt der Präsentation eines fertigen Modells, zeigen soll, was in deren eigener Modellierung problematisch sein könnte. Zunächst erfasst das System die Erfahrungen Studierender während der Erstellung eines Software-Designs. Das System verfolgt also die Aktivitäten der Modellierer und nutzt die Informationen dann zur Erstellung einer **Modellsammlung**. Bei Änderung eines Modells wird der Modellierer aufgefordert eine Status-Nachricht über die Änderung zu hinterlassen. Dadurch wurden Misconceptions Studierender erfasst und weiterverwertet. Es gibt **strukturiertes Feedback**. Dadurch, dass jede Änderung geloggt wird, können Nutzer über ein

Logfile und ein **Animationstool** sehen, wie sich ein Design entwickelt hat. Für das Feedback nutzt VorteX case-based reasoning(CBR). Studierende erhalten über das CBR-Modul strukturierte Ratschläge zu ihren Problemen. Dies ist möglich, da alle bisher erstellten Designs als sogenannte Fälle gespeichert werden. Der aktuelle Entwurf wird dann mit den anderen Fällen abgeglichen und dann relevanzbasiert ein ähnlicher Fall (über Äquivalenzzuordnungen) ausgewählt.

6.13.2 *Identifizierte Publikationen: Scaffolds zum Lernen von Programmieren und Programmiersprachen*

Interaktive Notizen, **eine Fallbibliothek**, **Problemlösende Aktivitäten** und ein **Kollaborationsmodul** haben Shabo et al. (1996) als Scaffolds für das Fach Computergrafik ausgewählt. Mittels interaktiver Notizen werden vor allem tiefergehende erklärende Seiten, so auch ein Glossar, verlinkt aber auch interaktive Komponenten, wie **Quizzes**, **Simulationen** oder **Animationen**. Die beiden anderen Module wurden auf Basis des Cognitive Apprenticeship Modells (Collins, 1987) erstellt und sind für Experten wie auch Novizen geeignet. Die Fallbibliothek wurde mit Hilfe von Experten auf Basis realer Erfahrung erstellt, weshalb Experten bei Problemen auf bestehende Fälle zurückgreifen können. Novizen profitieren von vereinfachten Fällen und Problemen, die in die Fälle der Experten mit integriert sind. Im Modul „Problemlösung“ wählt der Novize ein zu bearbeitendes Problem aus einem Repository, anschließend wird er aufgefordert, nach verschiedenen Lösungen zu suchen, indem er relevante Fälle, die in der Bibliothek gespeichert sind, abruft. Daneben erhält ein Lernender zusätzliches Scaffolding durch Vorschläge und Links zu bestehenden Notizseiten, Simulations- und Visualisierungstools um grundlegende Probleme zu verstehen, Expertensichten um das Problem zu analysieren und die korrekte Lösung zu finden sowie Hinweise zu Literatur zu Lösungsstrategien oder Lösungen, auch zu passenden existierenden Fällen. Um Kollaboration anzuregen wird auf ein existierendes Modul CaMILE eingesetzt.

Mit **ANNANN** wird ein Tool vorgestellt (White et al., 2006), das in der Lage ist den Studierenden Schritt für Schritt, **dynamisch den Entstehungsprozess einer Implementierung zu präsentieren**. Diese Funktion kann einerseits von einem Dozierenden verwendet werden, um den Studierenden zu demonstrieren, wie ein Programm entworfen wird und dabei zu erkennen, wie ein bestimmtes Programmierprinzip angewendet werden kann oder sie wird andererseits von den Studierenden angewendet, um das Problem eigenständig zu lernen.

Odisho, Aziz und Giacaman (2016) stellen **Visual Kinesthetic Pseudocode** (VKP) als eine Strategie vor, mit der Programmieranfänger beim Studium abstrakter und konzeptreicher Themen unterstützt werden können. Ziel der Autoren ist es, ein Sprungbrett zum Verstehen des Konzepts (d. h. des Denkprozesses) zu bieten, da dies die Voraussetzung für die eigentliche Codierung (d. h. die praktische Anwendung der Konzepte) ist. Die Komponente „Visuell“ soll verhindern, dass Novizen von Code abgeschreckt werden und sie statt textuell (durch die Programmiersprache) visuell in der Bildung ihrer mentalen Modelle zu unterstützen. Die Komponente „Kinästhetisch“ sorgt dafür Studierende kognitiv einzubinden und zu fordern, um deren Lernen effektiv zu machen. Zur Komponente Pseudocode schreiben die Autoren selbst: „For engagement to contribute to learning, it needs to be closely aligned to the ILO. The Fundamental Data Structures unit requires a high level of mastery that requires students to apply the thought process in manipulating data structures. This process, if documented as pseudocode, is a helpful precursor to writing“the real code"(Roy, 2006). The difference with VKP is that pseudocode is constructed by students visually and kinesthetically"(Odisho et al., 2016, S.928).

Boticki, Barisic, Martin und Drljevic (2012) stellen eine **Android App zum Lernen von Sortieralgorithmen** vor. Die Anwendung besteht aus vier Hauptkomponenten, dem Modul für interaktives Sortieren, einem Scaffolding-Modul, einem Motivationsmodul und einem Modul für die grafische Benutzeroberfläche. Das Scaffolding- Modul, das für die vorliegende Forschung von Interesse ist, bietet kontextbezogene textliche Hilfsmitteilungen mit Anweisungen zum Weitermachen oder zur Korrektur eines Fehlers.

Chalk (2000) stellt im Jahr 2000 grafische Werkzeuge für Softwaredesign und Softwaretest vor und plädiert für den Einsatz solcher Werkzeuge, wenn sie zusätzlich durch kollaborative Tools (wie einen Chat, Forum o.Ä.) unterstützt werden. Vorgestellt werden dann **JSP (Jackson Structured Programming)**, **JFP (Java Flowgraph Editor)** und **JAWAA (Java and Web-based Algorithm Animation)**, allesamt Tools um Codestrukturen zu visualisieren. Zusätzlich wurde ein Tool integriert um **Feedback** und ein **Forum** zu bieten. Wichtiger als eines der Tools von Chalk (2000) erscheint hier der Einsatz des zusätzlichen Werkzeugs für Feedback, da Studierende hier Austausch miteinander und mit ihrem Tutor hatten.

Linn und Clancy (1992) beschäftigt sich mit der Frage inwiefern **Case Studies**, zu Programmierproblemen in Pascal Studierende im Lernen der Programmiersprache bzw. dem Lösen von Programmierproblemen unterstützen können. Herausgefunden wurde, dass **Kommentare von Experten zu Entscheidungen und bestimmtem Vorgehen Studieren-**

den beim Verstehen der Probleme unterstützen können. Es hat sich herausgestellt, dass die zusätzliche Erläuterung des Experten mehr zum Verständnis beiträgt, als eine erarbeitete Lösung.

Linn (1992) stellen sogenannte **Hypermedia Tools** vor, die Studierende in der Programmierung mit Pascal oder Lisp unterstützen sollen. Dabei geht es vor allem darum, Programmierwissen zu strukturieren: Zum einen soll Studierenden gezeigt werden, wie Experten ihr Wissen organisieren und darum sie zu motivieren ihr eigenes Wissen zu strukturieren. Einen Weg sieht Linn (1992) darin, Studierende mit einer **Bibliothek an existierenden Beispielen und Templates** zu unterstützen, da angenommen wird, dass Studierende zu viel Zeit damit verbringen, diese Beispiele und Templates zu suchen. Noch mehr profitieren Novizen von Templates, wenn diese im Rahmen von Fallstudien vorgestellt bzw. eingeführt werden. Eine weitere Möglichkeit der Unterstützung wird in **zusätzlichen Kommentaren** und der Organisation derer gesehen, weshalb die entwickelte Bibliothek um diese Funktion erweitert wurde.

In der Arbeit von Matthews, Hin und Choo (2015) werden zwei Scaffolds für eine Programmierveranstaltung evaluiert: Sogenannte **Review-Fragen** und **Advanced Organizer**. Motiviert wird die Arbeit darüber, dass nicht alle Studierenden mit dem gleichen Vorwissen die Veranstaltung besuchen und deshalb geeignete Scaffolds gefunden werden müssen, Studierende dabei zu unterstützen, vorhandenes Wissen mit neuem Wissen korrekt in Verbindung zu setzen. Beide Hilfsmittel sollen also Vorwissen aktivieren. Review-Fragen werden textuell präsentiert und sollen Lernern helfen sich daran zu erinnern, wie Programmkontrollstrukturen funktionieren. Studierende sollten die Ausgabe eines Algorithmus finden, der die Kontrollstruktur anhand eines Satzes vorgegebener Daten demonstriert. Das Content Object als Advanced Organizer wurde verwendet noch bevor der eigentliche Programmieranteil der Veranstaltung begann. Die Content Objects waren eine kleine Einheit, um den Fluss der Programmierstruktur mit Hilfe von Bildern und Animationen zu erklären. Studierende mussten die Animationsschaltfläche bedienen, um die Ausgabe des Algorithmus zu verstehen. Ein Content Object ist ein kleines Stück Inhalt, das ein einzelnes Lernergebnis erzielt, um die Abstraktion der Programmierung zu erleichtern. Ein Vergleich der Review Fragen mit den Content Objects ergab, dass Studierende zweiteren Scaffold bevorzugen. Betrachtet man die Performance der Studierenden nach den beiden Scaffolds, unterschieden sich die beiden nicht.

6.13.3 *Identifizierte Publikationen: Scaffolds für den gesamten Softwareentwicklungsprozess*

Ludi, Natarajan und Reichlmayr stellen in ihrer Arbeit (2005) die Umstrukturierung eines Software Engineering Kurses auf ein projektorientiertes Format vor. Dabei beschreiben sie verschiedene Scaffolds als „in-class activities", die sie für jeden Inhalt speziell ausgewählt haben. Bei der Auswahl beschreiben sie kein spezielles Vorgehen, die Vorlesungsinhalte, zu denen die Aktivitäten erstellt wurden, entsprechen dem Software-Lebenszyklus. Nach einführenden **Hands-On Tutorials** zur Entwicklungsumgebung, Konfigurationsmanagement und einem Testframework werden zum Thema **Requirements Engineering Anforderungen aufgestellt und bestehende Anforderungen verbessert**. Für Zustandsdiagramme wurden auch **Übungen** durchgeführt und zum Thema Design Pattern wurde **mit Hilfe von Beispielen im Rahmen eines gemeinsamen Work-Throughs** gearbeitet. Außerdem wurden für die Testphasen und **rollenbasierte Inspektionsmeetings** durchgeführt. Daneben wurden noch **teamprägende Aktivitäten**, **Klassendiskussionen** und **Fallstudien** eingeführt. Zum Abschluss des Kurses und zur Klausurvorbereitung wurden die Studierenden gebeten **Klausurfragen** zu erstellen.

Higgins, Mtenzi, O'Leary, Hanratty und McAvinia (2017) stellen ein Framework vor, das im Kern deklaratives und prozedurales Wissen von Novizen während des gesamten Softwareentwicklungsprozesses unterstützen soll. Das **Unterstützungs-Tool** bietet Studierenden die **Plattform**, auf Probleme gelöst werden können, sowie **diagrammatische Werkzeuge zur Unterstützung ihrer Analyse-, Design- und Reflexionsarbeit**. Das Support-Tool zur Problemlösung ist auf Polya's 4-Stufen-Modell (Polya (1985)) aufgebaut: Verstehen des Problems, Ableitung der Aufgaben, Design und Code, Evaluation der Lösung und Lernen. In Stufe eins werden Lernende mit Hilfe des Unterstützungsinstruments gebeten, ihr **Verständnis** entweder für ein Problem zu **artikulieren**, das sie bereitgestellt haben, oder für ein Problem, das ihnen als Teil der Lernprozessphase zur Verfügung gestellt wird. In Stufe zwei unterstützt das Tool die Studierenden beim Brainstorming von möglichen Aufgabenkandidaten mit Hilfe einer **Mind Map**, bei der die Zusammenfassung des Problems die zentrale Aufgabe ist. In Stufe drei geht es um die Umsetzung. Dort **erleichtert** das Tool den **Wechsel zwischen den in der Mindmap identifizierten Aufgaben, den verbundenen Designs und Codes**. In der letzten Stufe fordert das Tool die Lernenden auf, **Bewertungskriterien anzuwenden**, um zu evaluieren, ob ein Lösungsansatz von der Erfahrung im Umgang mit vorherigen Problemen profitiert hat.

Gašević, Adesope, Joksimović und Kovanović (2015) beschreibt Ansätze, wie **Online-Diskussionen** zwischen Studierenden mit hoher kognitiver Präsenz gestaltet werden können. Insbesondere wird in diesem Papier vorgeschlagen, dass ein hohes Maß an kognitiver Präsenz in Online-Kursen auf der Grundlage des Community of Inquiry-Modells erleichtert werden kann: Zum Einen durch selbstreguliertes Lernen durch extern unterstütztes Scaffolding und zum Anderen durch computergestütztes **kollaboratives Lernen mit Hilfe von Rollenzuweisungen**. Als sozialkonstruktivistisches Modell befasst sich das Community of Inquiry-Modell mit Lernen höherer Ordnung - einem Ideal der Hochschulbildung - durch computervermittelte Interaktion von Lernenden und Lehrenden (z. B. Garrison, Anderson und Archer (1999), Garrison, Anderson und Archer (2001)). Laut Garrison et al. (2001) beinhaltet eine Community of Inquiry die (Neu-) Konstruktion von Erfahrung und Wissen durch die kritische Analyse des Lernstoffs, das Hinterfragen und das Infragestellen von Annahmen (Garrison et al., 2001, S.7). Es konnte gezeigt werden, dass Scaffolding über die Diskussionsguidelines einen höheren Effekt auf die kognitive Präsenz hatte als extrinsisch induzierte Motivation, wie beispielsweise Noten. Kognitive Präsenz ist dabei definiert als, "the extent to which the participants in any particular configuration of a community of inquiry are able to construct meaning through sustained communication (Garrison et al., 1999, S.89). Untersucht wurde ein Software Engineering Kurs im Master. Im Rahmen eines Kontrollgruppenexperiments wurden beiden Gruppen Guidelines zur Beteiligung an einer Online Präsentation, eingereicht durch Kommilitonen, zur Verfügung gestellt. Die Studierenden hatten die Aufgabe auf Basis dieser Präsentation, Themen zu diskutieren. Den Studierenden wurden dabei zwei Rollen zugewiesen, die des Experten, der die Präsentation eingereicht hat und demnach fachlich verteidigt und andererseits, die des praktizierenden Forschers, der sich aktiv anderen Präsentationen zuwendet und sich an Diskussionen zu diesen beteiligt.

6.13.4 *Identifizierte Publikationen: Scaffolds zum Lernen von (Software-) Projektmanagement*

Espinosa, Noguez, Benes und Bueno (2005) untersuchen ein webbasiertes Projektmanagement-Tool als Scaffold in einem Kurs zum Thema Compilerbau. Dieses vereint **kollaboratives Arbeiten** mit einem **Tutorensystem** und gleichzeitigem **Inhaltsmanagement** um Studierenden die fachliche, soziale, kognitive Relevanz der Themen langfristig zu vermitteln. Das System beinhaltet folgende Komponenten: Einen **Einsatzplaner**, einen **Task-Recorder**, ein **Überwachungs-Modul**, einen **Generator für Berichte**,

ein **Benutzermodell**, ein Modul zur **Datenbankverwaltung**. Konkret werden die Studierenden bei der Erstellung eines Projektportfolios unterstützt, das folgende Komponenten enthält:

- Einstellungen und vertragliche Vereinbarungen
- Systemanalyse und Designkomponenten
- Integrations- und Wiederherstellungseinheiten
- Rekapitulations- und Umsetzungsaktionen

Die ersten drei Einheiten des Kurses sollen den Studierenden vor allen einen Einblick geben, was sie im Rahmen von vertraglichen Vereinbarungen versprechen und wie sie eine Projektgröße bestimmen können. Alle Einheiten werden in einer Kombination aus Vorlesung und Workshops durchgearbeitet, Theorie und Praxis beleuchtet.

Die Publikation von Papadopoulos, Demetriadis und Stamelos (2007) untersucht Unterstützung im Bereich der Lehre zu Softwareprojektmanagement. Sie untersuchen sogenannte **„Question Prompts"** als kognitive Unterstützungsmöglichkeit. Die Autoren argumentieren, dass eine durch dedizierte Fragestrategie Lernende dazu veranlasst werden können, sich auf wichtige kontextbezogene Informationen zu konzentrieren, die im Material enthalten sind. Solche Fragen können so gestaltet werden, dass sie die kontexterzeugenden kognitiven Prozesse der Lernenden aktivieren und sie so anleiten, sich des Kontextes einer Situation bewusst zu werden.

Hislop und Ellis (2009) betrachten Kommunikationsfähigkeiten sowohl schriftlich wie mündlich als zentral für Software Engineering Studierende und setzen dabei auf die Arbeit mit **Templates** in projektbasiertem Arbeiten. Gerade die Art zu Formulieren, zu Kommunizieren und Phasen des Softwarelebenszyklus zu dokumentieren stehen hierbei im Vordergrund.

6.13.5 *Identifizierte Publikationen: Scaffolds zum Lernen von Requirements-Engineering*

Haidry, Falkner und Szabo (2017) identifizierten in ihrer Arbeit kognitive Software Engineering- spezifische Strategien und haben deren Wirksamkeit mit Hilfe eines Scaffolding Konzepts zur Anwendung dieser Strategien geprüft. Da verschiedene Phasen des Softwarelebenszyklus unterschiedliche kognitive und metakognitive Fähigkeiten erfordern können, wurden die identifizierten Strategien entsprechend den Phasen im Softwareentwicklungsprozess ausgerichtet, um auch aufgabenbasierte SE-spezifische Strategien zu identifizieren. Für das Strategietraining wurde ein **computerbasiertes animiertes Tool** entwickelt. Leider werden ausschließlich für die Anforderungsanalyse vier konkrete Strategien

präsentiert: **Objekte, Dateneingaben und -ausgaben auflisten**, **korrespondierende Objektfunktionen und -attribute auflisten**, **Benutzer- und Systeminteraktionen bestimmen**, **Spezifische Verbalphrasen auf Funktionen abbilden**. Weiterhin wurde **Concept Mapping** als allgemeine Visualisierungsstrategie identifiziert, konkret, z. B. Interaktionen zwischen Entitäten und Klassenobjekten, um das Design zu verdeutlichen. Für die Designphase wurde die geringste Anzahl an Strategien identifiziert, diese wurden auch nicht beschrieben. Die Empfehlung ist, diese Strategien konkret in SE-Kursen zu integrieren, sodass Studierende ihre kognitiven und metakognitiven Fähigkeiten verbessern können.

6.13.6 *Identifizierte Publikationen: Scaffolds zum Lernen von Datenbanken*

In der Arbeit von Murray, Ryan und Pahl (2003) liegt der Fokus auf Studierenden, die den Umgang mit Datenbanken lernen sollen. Sie stellen ein Lerntool für Datenbanksysteme vor und stützen sich beider Auswahl der Scaffolds auf die Unterscheidung von zwei Wissenstypen: deklaratives Wissen und prozedurales Wissen. Sie bieten **visuelle** und **auditive** Scaffolds in Form von **Vorlesungsvideos** an, die aber von Studierenden gesteuert werden können. Weiterhin gibt es **animierte Bilder** und **Simulationen** von Schlüsselelementen in der Datenbankumgebung. Für die Laborphase und die Assessmentphase gibt es **interaktive Webtechnologien wie Java Applets**.

6.14 ZUSAMMENFASSUNG

Überblickt man das Feld der existierenden Arbeiten im Bereich Software Engineering, wie in FF2.1 formuliert, so kann man hier ein breites Feld an den verschiedensten Scaffolds verzeichnen. Es gibt visuelle und auditive Scaffolds, Feedback in verschiedenen Varianten, Mindmapping in verschiedenen Varianten, Eingabeaufforderungen in verschiedenen Varianten, dem Lernen von Experten, der Bereitstellung von Bibliotheken mit Beispielen und/oder Templates und einige Eigenentwicklungen beispielweise zu kollaborativem Arbeiten. Es lässt sich deshalb keine eindeutige Strategie zur Konzeption von Unterstützungsmaßnahmen ableiten. Nichtsdestotrotz wurde die Arbeit von den bestehenden Ansätzen inspiriert: Versionierbare Artefakte, Kommentarfunktionen und die Bereitstellung von existierenden Projekten als Sammlung bzw. Nachschlagewerk für Studierende sind auf Basis der existierenden Publikationen realisiert worden. Damit ist Forschungsfrage 2.1. (FF2.1) aus Sicht der Literatur beantwortet.

Zudem wurden im Rahmen von drei Experimenten Scaffolds evaluiert, die aktuelle Technologien verwendeten und damit auch Beiträge zur aktuellen Lehr-Lernforschung leisteten, wie in FF2 und zugehörigen Unterfragen aufgeworfen. In diesen Experimenten wurden Parameter wie Motivation, Lernerfolg aber auch der Cognitive Load gemessen. Die Ergebnisse sind erfolgsversprechend, müssen aber in weiteren Experimenten und im Feldeinsatz evaluiert und weiterentwickelt werden. Zudem ist hier auch der Novelty-Effekt miteinzubeziehen, der besagt, dass bei der Einführung einer neuen Technologie bei Lernern zunächst Faktoren der Verbesserung der Performanz, gesteigertes Interesse und gesteigerte Motivation zu verzeichnen sind, die aber tendenziell wieder abnehmen, wenn die Lerner mit der Technologie vertrauter werden (Metcalf et al., 2019).

Die Integration der Problemerkennung (Abschnitt 6.5) bietet Dozierenden die Möglichkeit einzelne Scaffolds auf Basis des detektierten Problems zu empfehlen. Außerdem können Studierende wie Dozierende gleichermaßen einzelne Hilfsmittel bewerten (siehe Abschnitt 6.5). Mit Hilfe dieser Ansätze wurden mögliche Antworten für FF2.3 vorgestellt.

Die Parameter Usability, Performanz und Motivation der Studierenden zeigen eine mögliche Eignung von Scaffolds. Weitere Parameter, wie beispielweise die Nutzungszeit, die Häufigkeit der Verwendung oder auch die vergebenen Bewertungen, könnte ein Dozierender heranziehen, um detailliertere Ergebnisse über die Nutzung der Scaffolds zu erhalten. In Abschnitt 7.3.5 wird weiter beschrieben, wie innerhalb eines Softwaresystems zusätzliche Daten zur Nutzung und Nutzungsdauer erfasst werden können.

7

ENTWICKLUNG: ARCHITEKTUR & IMPLEMENTIERUNGSKONZEPTE

„To devise a plan, to conceive the idea of the solution is not easy. It takes so much to succeed; formerly acquired knowledge, good mental habits, concentration upon the purpose, and one more thing: good luck. To carry out the plan is much easier; what we need is mainly patience."

— George Pòlya (Polya, 1985, S. 26)

Bereits in Abschnitt 5.3.5 und besonders in Tabelle 5.33 wurden Anforderungen beschrieben, die laut Hevner et al. (2004) auch als Bedarf oder Problem bezeichnet werden können. Sobald diese Anforderungen vorhanden sind, wird die Forschung in zwei sich ergänzenden Phasen vollzogen, die sich beide im Block des Design Science Research wiederfinden: die Erzeugung oder Konstruktion und die Rechtfertigung oder Evaluation. Im vorherigen Kapitel 6 wurden verschiedene Inkremente konzeptioniert und zum Teil evaluiert. In Kapitel 6 finden sich ebenfalls High-Level Anforderungen und konzeptionelle Designentscheidungen an eine zu entwickelnde Software. Es entspricht einem Teil des Designzyklus und wird nun im folgenden Kapitel (Abschnitt 7.1) fortgesetzt. Als zentrales Inkrement wird die Ariadne Software vorgestellt (Abschnitt 7.1.1), die Scaffolds aus dem vorherigen Kapitel zusammenführt (Abschnitt 7.2.4). Es werden architektur- und forschungsrelevante Komponenten der Software beschrieben. Aus Gründen des Umfangs werden nicht alle einzelnen Komponenten diskutiert (Abschnitt 7.3), sondern lediglich zentrale Repräsentanten pro Kapitel dargestellt, ohne eine Reduktion der Diskussion der wissenschaftlichen Fragestellungen vorzunehmen.

7.1 VORÜBERLEGUNGEN UND HINFÜHRUNG

Um alle Anforderungen, die aus den vorherigen Kapiteln hervorgegangen sind, erfüllen zu können, wurde eine eigene Softwarelösung namens

Ariadne[37] entwickelt, die sowohl dem Wunsch der Studierenden nach einem „anderen Programm“ (siehe Abschnitt 6.2) nachkommt als auch Scaffolds integriert. Im Bereich der Programmierausbildung sind bereits Ansätze für Einsteigerumgebungen und zum Teil integrierte Hilfen und eine Möglichkeit diese zu evaluieren, bekannt (z. B. Ebert, 2018; Mohammed & Abdelazziz, 2013). Für die Modellierung mit der UML ist dies nicht der Fall. Die bisher bekannten eingesetzten Tools zur Modellierung[38] besitzen keine integrierten Scaffolds, wie sie hier konzipiert wurden.

Ariadne bietet Nutzern diverse grundlegende Funktionen an, die so angelegt sind, dass sie in Analogie ((Sutch, 2007); siehe Abschnitt 6.2) zu bereits bekannten Werkzeugen verwendet werden können. Aus Sicht von Lehrenden stellt Ariadne verschiedene Kategorien von Scaffolds bereit, deren Inhalte, da sie immer an die Studierendengruppe angepasst sind, auch selbst bestimmt werden können. Diese Inhalte können außerdem bewertet werden. Lehrende haben die Möglichkeit, selbst Regeln zu definieren, auf die Ariadne mit entsprechenden Meldungen und der Empfehlung von Scaffolds reagiert. Das Logging einzelner Parameter der Umgebung ermöglicht Lehrenden zudem Rückschlüsse bezüglich der Verwendung der Scaffolds. Lernende nutzen die Modellierungsumgebung zur aufgabenbasierten Modellierung von UML-Diagrammen und haben die Möglichkeit, innerhalb der Umgebung Scaffolds zu verwenden. Sie müssen keine Medienwechsel während der Modellierung vollziehen. Vorteilhaft ist, dass von Dozierenden geeignetes Material zur Verfügung gestellt werden kann. Auch Lernende können Scaffolds bewerten und im Rahmen der Tutorials selbst eigene Recherchen und Tutorials hinzufügen, die sie für gut befinden. Die Wertung eines Scaffolds ist für alle Nutzer sichtbar.

Wie bereits in Kapitel 2 vorgestellt, hat Ariadne folgende zentrale Komponenten:

1. Editoren zur Modellierung mit der UML
2. Scaffolds und deren Empfehlungen
3. Logging

37 In der Sage von Theseus, der griechischen Mythologie ist Ariadne die Tochter des Königs Minos, die Theseus, für den Weg durch ein Labyrinth zu Minotauros (einem zwitterhaften Geschöpf, das halb Mensch und halb Stier war), einen Knäuel Faden reichte. Das Ende des Fadens solle er am Tor des Labyrinths festknüpfen und ablaufen lassen bis er zu Minotauros gelangte. Nachdem Theseus den Minotauros getötet hatte, konnte er entlang des Fadens das Labyrinth wieder verlassen (Schwab, 2006, S. 147). Der Faden als Grundmotiv hat sich weiterentwickelt und wird als leitender Gedanke, Weg oder Richtlinie mit dem „roten Faden“ verknüpft (siehe z. B. von Goethe, 1810).

38 Ein Überblick über existierende Tools findet sich unter: https://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools

7.1.1 *Architektur*

Auf Basis der erhobenen konzeptuellen Anforderungen wurde die Systemarchitektur entworfen. Das Ergebnis wird nun einerseits schematisch in der Architekturübersicht und andererseits mit einem UseCaseDiagramm sowie dem Model-View-ViewModel (MVVM) Architekturmuster visualisiert.

7.1.1.1 *Architekturrelevante Use-Cases*

Im vorherigen Kapitel wurden bereits Anforderungen an eine Software vorgestellt. Nun wird das architekturrelevante Use-Case-Diagramm präsentiert (Abb. 7.1). Es gibt zwei primäre Akteure Dozierende oder Studierende, die Nutzer des Systems Ariadne darstellen. Dozierende haben erweiterte Rechte und können, im Gegensatz zu Studierenden, Scaffolds zusätzlich ändern und bereitstellen, während Studierende Scaffolds ausschließlich verwenden und bewerten können. Die weiteren Use-Cases triggern alle Nutzer auf identische Weise, weshalb die primären Akteure generalisiert wurden, zu einem Akteur „Nutzer“.

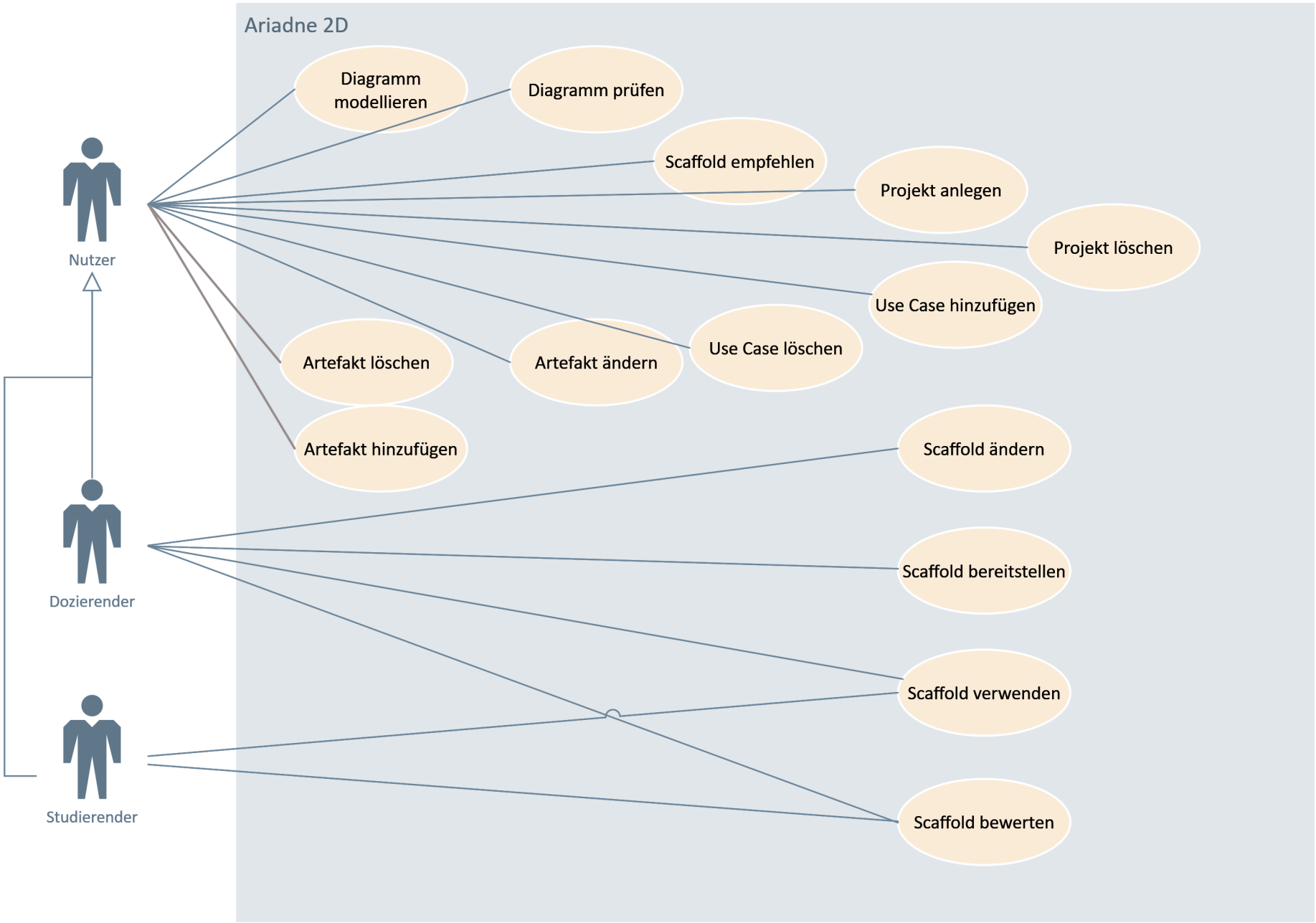


Abbildung 7.1: Use-Case-Diagramm der architekturrelevanten Use-Cases von Ariadne

7.1.1.2 *Architekturübersicht*

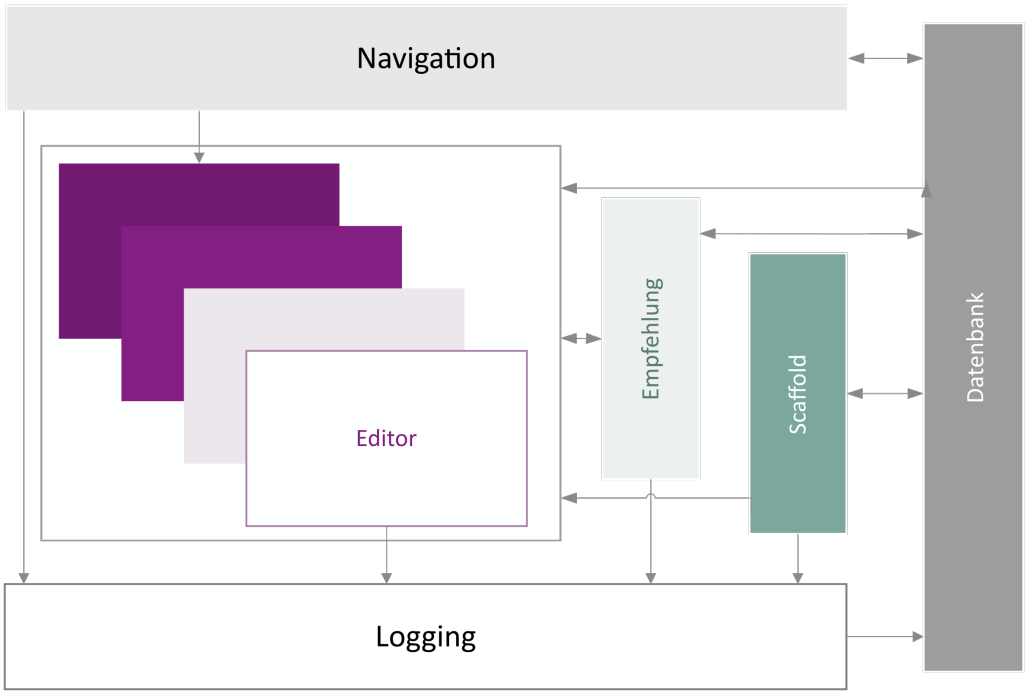


Abbildung 7.2: Übersicht über die Architektur

Im Wesentlichen lassen sich für die Architektur folgende Komponenten identifizieren (siehe Abb. 7.2):

- Es gibt drei Datenbanken, die später noch näher erläutert werden (siehe Abschnitt 7.1.2). Auf diese greifen alle Applikationen lesend und schreibend zu.
- Es gibt Editoren zur Modellierung und zur Erfassung von Text.
- Zudem gibt es unterstützende Scaffolds.
- Daneben gibt es die Komponente, die die Logik zu den Empfehlungen enthält.
- Außerdem gibt es eine Navigationsstruktur zur Nutzung der einzelnen Komponenten.
- Weiterhin gibt es ein Logging-System, das verschiedene Nutzerdaten loggt.

7.1.1.3 *Model View ViewModel-Muster*

MVVM ist eine Variante des Model View Controller (MVC) Musters. Es gehört zu den Mustern, die „für Benutzeroberflächen und zur Interaktion mit dem Anwender eingesetzt werden“ (Geirhos, 2015, S. 47). Zudem

wird es als Verhaltensmuster klassifiziert. Es „ist vor allem bekannt geworden durch Windows Presentation Foundation (WPF) und Silverlight, aber auch durch HTML5. Im Kern geht es dabei um die Trennung der Verantwortlichkeiten zwischen den Entwicklern und den Designern einer Anwendung“ (Geirhos, 2015, S. 47). Es bedeutet dies „die Trennung zwischen Darstellung, (Geschäfts-)Logik und Benutzerschnittstelle“ (Geirhos, 2015, S. 549). Geirhos (2015) empfiehlt bei der Arbeit mit WPF die Verwendung dieses Musters. Auch in Ariadne wird dies eingesetzt. In Abb. 7.3 wird das MVVM-Muster schematisch dargestellt.

„Eine WPF-Benutzeroberfläche wird mithilfe der Auszeichnungssprache Extensible Application Markup Language (XAML) deklariert, die aber keinen Code beinhaltet“ (Geirhos, 2015, S. 549). Deshalb sind die Codebestandteile in die Datei hinter der XAML-Datei ausgelagert und die Darstellung der Oberfläche von dem eigentlichen Code getrennt (Geirhos, 2015).

In Ariadne ist das MVVM-Muster folgendermaßen (siehe auch Abb. 7.3) umgesetzt:

- Model: CS-Dateien
- View: XAML-Dateien
- ViewModel: XAML.CS- Dateien

Abbildung 7.3: Das MVVM-Muster

MODEL „Das Model hat [...] bei diesem Muster vorrangig die Aufgabe der Datenhaltung (Abb. 7.3, Datenbankzugriff). In manchen Implementierungen ist das die einzige Aufgabe. Um alle weiteren Aufgaben kümmert sich [...] das ViewModel“ (Geirhos, 2015, S. 550). Das Model kennt seine Views nicht und beinhaltet keinen Code, der die Darstellung beeinflusst. In Ariadne wird innerhalb des Models die Datenbankzugriffsschicht realisiert. QC 1 zeigt beispielhaft ein *Model* das Felder für die Verwaltung von Literatur verwaltet.

```
public class Literature : BindableBase, INotifyPropertyChanged,
↪  IChildNodesSelector{
    /// <summary>
    /// LiteratureName as shown in the literature tree
    /// </summary>
    public string LiteratureName { get; set; }

    /// <summary>
    /// a short description for this literature
    /// </summary>
    public string LiteratureDescription { get; set; }

    /// <summary>
    /// source where to find a word
    /// </summary>
    public string LiteratureSource { get; set; }
    ...
```

QC. 1: *Model* des MVVM-Musters beispielhaft für Literatur

VIEW Die *View* besteht aus dem *XAML*-Code zur Beschreibung der Benutzeroberfläche. Dazu gehören alle Steuerelemente und die zugehörigen Eigenschaften, aber auch die gezeichneten Elemente (Geirhos, 2015, S. 551). Abbildung 7.4 zeigt exemplarisch die angelegten XAML-Dateien und die dahinterliegenden *XAML.CS*-Dateien.

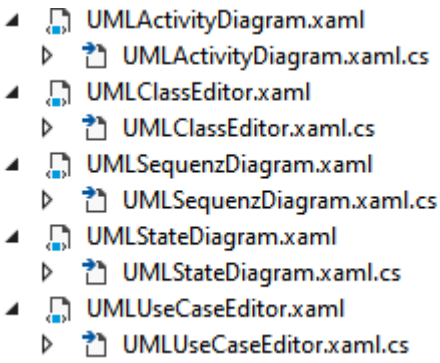


Abbildung 7.4: Beispiele für XAML und CS-Dateien der Editoren in Ariadne

Die *View* kann aber auch weitere Aufgaben übernehmen:

- „Formatierung der Anzeigedaten“ (Geirhos, 2015, S. 551): Beispielhaft wird in QC 2 die Anzeige formatiert nach Name (*LiteratureName*), einer Kurzbeschreibung (*LiteratureDescription*) und einer URL (*LiteratureSource*) gezeigt.

- „Datenbindung, also [...] das Abrufen der Daten aus dem Model und die Darstellung in dem gebundenen Steuerelement“ (Geirhos, 2015, S. 551). *Literaturename, LiteratureDescription, LiteratureSource* werden in der Klasse *Literature.cs* (siehe QC 1) gesetzt und in QC 2 über das Keyword *„Binding“* an das Steuerelement gebunden (Abb. 7.3, Bindung und Ausgabe der Felder der Datenbank).
- „Ereignisbehandlung, also zum Beispiel das Reagieren auf einen Tastenklick oder das Ausklappen von Baumknoten“ (Geirhos, 2015, S. 551): Hier sei beispielhaft das Ereignis *„BSearchContent_OnClick SearchContent_Click“* in der View (QC 3) genannt.
- „Behaviors, gemeint sind solche zum Abbilden von komplexeren Benutzerinteraktionen, beispielsweise das Neupositionieren von Steuerelementen, Zooms oder Drag-and-Drop“ (Geirhos, 2015, S. 551). Diese Aufgabe trifft in der Implementierung von Ariadne nicht zu.

```
<TextBox Grid.Row="0" Grid.Column="0"
 ↪  Grid.ColumnSpan="3" Padding="2,1"
 ↪  Text="{Binding LiteratureName}"
HorizontalAlignment="Left"
 ↪  Background="Transparent" FontWeight="Bold"
 ↪  FontSize="14" Foreground="Blue"
 ↪  Margin="2,0,0,0"
BorderThickness="0" BorderBrush="Transparent"
 ↪  TextWrapping="Wrap" IsReadOnly="True"
 ↪  IsHitTestVisible="False"/>
<TextBox Grid.Row="1" Grid.Column="0"
 ↪  Grid.ColumnSpan="3" Padding="2" Text="{Binding
 ↪  LiteratureDescription}"
HorizontalAlignment="Left"
 ↪  Background="Transparent" Margin="2,2,0,0"
BorderThickness="0" BorderBrush="Transparent"
 ↪  TextWrapping="Wrap" IsReadOnly="True"
 ↪  IsHitTestVisible="False"/>
...
```

QC. 2: Bindung eines Objekts an ein Steuerobjekt, z. B. Binding LiteratureName an Textbox

```
<Label x:Name="labelSearchContent" Content="Suchen" Grid.Row="1"
 ↪  Margin="10,5,5,0" BorderBrush="Black" FontSize="16"
 ↪  Grid.Column="0" />
<TextBox x:Name="textboxOnlineSearch" Grid.Row="1" Grid.Column="1"
 ↪  Margin="5"
Grid.ColumnSpan="2" Background="White" BorderBrush="Gray"
 ↪  Height="28" TextWrapping="NoWrap" BorderThickness="0"
 ↪  KeyUp="TextboxSearchContent_KeyUp"
 ↪  VerticalContentAlignment="Center" />
<Button x:Name="buttonSearchContent" Grid.Row="1" Grid.Column="3"
 ↪  HorizontalAlignment="Stretch" Height="30" Margin="5"
 ↪  Content="Suchen" Click="
 ↪  BSearchContent_OnClickSearchContent_Click"/>
...
```

QC. 3: Ereignisaufruf in Tutorials beim Absetzen der Suche

VIEWMODEL Das *ViewModel* bringt *View* und *Model* zusammen, es werden das *XAML*, das die View repräsentiert und die *CS*-Klasse, die das *Model* darstellt, zusammengeführt und koordiniert. „Das *ViewModel* enthält die komplexeren und abstrakteren Operationen einer *View* und auch den Zustand der *View*, sowie all die Eigenschaften und Methoden, die für die Kommunikation zwischen *View* und *ViewModel* einerseits und *ViewModel* und *Model* andererseits notwendig sind“ (Geirhos, 2015, S. 551). In der Ariadne-Architektur beinhaltet das ViewModel im Allgemeinen die Logik, die außerhalb der Datenbankkommunikation stattfindet.

"Der Kommunikationsablauf zwischen View und Model, der im *ViewModel* verarbeitet wird, wird typischerweise so beschrieben:

- Eine *View* nimmt ein Ereignis entgegen.
- Das *Ereignis* wird einen Methodenaufruf im *ViewModel* auslösen, das diese Aktion wiederum an das *Model* weiterleitet und anschließend die *View* auffordert, sich zu aktualisieren"(Geirhos, 2015, S. 552).

Für die Kommunikation zwischen *View* und *ViewModel* gibt es laut Geirhos (2015) folgende Möglichkeiten:

- die eben erwähnten Ereignisse: In QC 4 ist exemplarisch ein Auszug aus dem *ViewModel* für den Use-Case-Diagramm-Editor dargestellt, das ein mögliches Ereignis behandelt. Es findet hier ein Logging statt und keine Weiterleitung an das zugehörige *Model*. In QC 5 ist exemplarisch ein Auszug aus dem *ViewModel* für die Textanalyse

dargestellt. Bei dem Event *Item_Click* wird an das Model *UseCaseAnalysisResult* benutzt, um den ausgewählten Use-Case aus der Liste von Use-Cases zu entfernen.

- Datenbindung: Anknüpfend an das vorherige Beispiel (QC 5) werden über *„listViewUseCases.ItemsSource= UseCases“* die Liste der Use-Cases aus dem ViewModel an die Liste der Use-Cases in der *View „listViewUseCases“* gebunden.
- Message Queuing: Diese Kommunikationsart wird in Ariadne nicht verwendet.

```
private void DiagramControl_ItemCreating(object sender,
↪  DiagramItemCreatingEventArgs e){
        Logger.Write("Use Case Diagram: item created",
        ↪  ExceptionCategory.Recommendation,
        ↪  ExceptionPriority.Normal,
        ↪  ExceptionID.ExceptionStartID,
        ↪  TraceEventType.Information, "Use Case Diagram");
        ...
        if (e.Item != null)
                DiagramDesignerControlEx.AriadneLogger("Use Case
                ↪  Diagram: ItemCreating - " + AriadneTelemetry.S
                ↪  erializeJson(e.Item.CustomStyleId.ToString(),
                ↪  e.Item.Position, e.Item.RenderSize),
                ExceptionCategory.EyeTracker,
                ↪  ExceptionPriority.Normal,
                ↪  ExceptionID.ExceptionStartID,
                ↪  TraceEventType.Information, "Use Case
                ↪  Diagram");
        ...
```

QC. 4: Auszug aus dem *ViewModel* des MVVM-Musters für den Use-Case-Diagramm-Editor, das ein mögliches Ereignis behandelt.

```
private void BiRemoveUseCase_ItemClick(object sender,
↪  ItemClickEventArgs e){
    if (listViewUseCases.Items.Count > 0){
        var useCase =
        ↪  listViewUseCases.Items[listViewUseCases.SelectedIndex]
        ↪  as UseCaseAnalysisResult;
        UseCases.Remove(useCase);
        ...
    }
```

QC. 5: Auszug aus dem *ViewModel* des MVVM-Musters für die Textanalyse, das ein mögliches Ereignis behandelt. Bei dem Event *Item_Click* wird das Model *UseCaseAnalysisResult* benutzt, um den ausgewählten Use-Case aus der Liste von Use-Cases zu entfernen.

Mit Hilfe des MVVM-Patterns kann die prinzipielle Architektur und Funktionsweise Ariadne vollständig beschrieben werden. Es wurden lediglich Ausschnitte gezeigt, da die Funktionsweise für die übrige Software analog funktioniert.
Nach der Beschreibung der Architektur folgt nun die Darstellung des Datenbankmodells. Innerhalb der untersten Ebene in Abb. 7.2 sind drei Datenbanken angesiedelt.

7.1.2 *Datenbankmodell*

- Die Datenbank *AriadneDB* kümmert sich um die zentrale Vorhaltung aller Artefakte der Umgebung und um die Nutzerverwaltung[39].
- Die *AriadneRecommendation*-Datenbank verwaltet den abgeleiteten Problemkatalog (siehe Abschnitt 5.3.5) und zugeordnete Empfehlungen für Scaffolds.
- Die Datenbank *AriadneTelemetry* speichert die Log-Daten, die pro Nutzer gelogged werden können.

Gemeinsame Informationsbasis über alle Datenbanken hinweg sind die Tabellen *User*, *Projekt* und *Use-Case*. Es werden im Folgenden zentrale Tabellen der Datenbanken näher erläutert.

7.1.2.1 *AriadneDB*

39 Nutzer werden anonymisiert abgespeichert. Im Login-Dialog zu Ariadne wird auf die Speicherung der Daten hingewiesen. Wenn Nutzer dies nicht akzeptieren, können keine individuellen Empfehlungen gegeben werden.

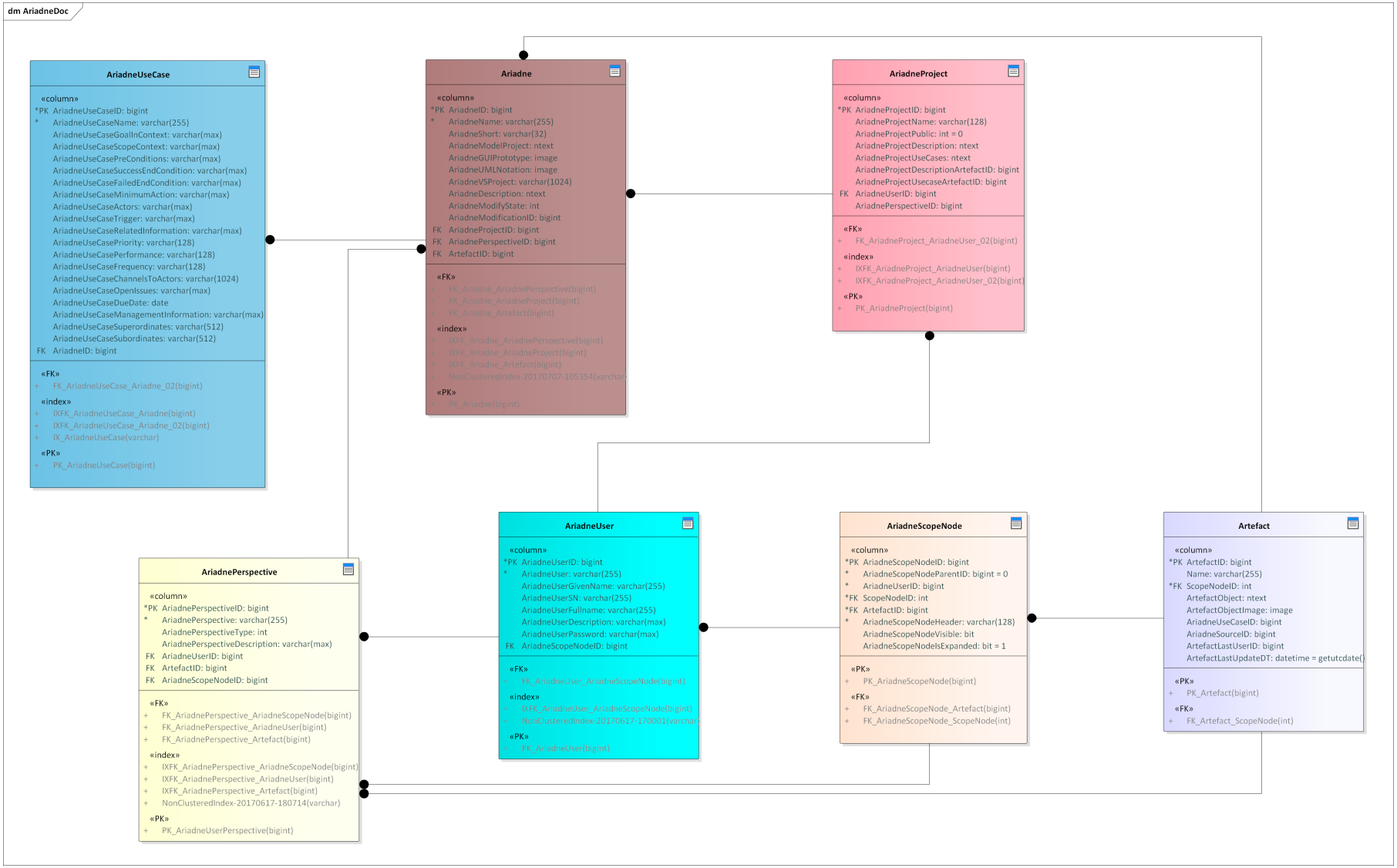


Abbildung 7.5: Vereinfachtes Datenmodell der AriadneDatenbank

Das Datenmodell, deren Strukturen sich im Wesentlichen in die interne Datenhaltung und die Benutzersteuerung aufteilen lassen, ist in Abb. 7.5 abgebildet. Alle Tabellen, die nicht die logische Struktur beschreiben oder technische Tabellen darstellen, sind hier aus Gründen der Übersichtlichkeit entfernt[40].

Die Tabelle *AriadneScopeNode* verwaltet die Navigation. Die übrigen Tabellen werden für die interne Datenhaltung benötigt. Fokussiert wird im Folgenden die Nutzersteuerung. Die Entitäten der *AriadneScopeNode* Tabelle werden mit Referenzen auf die Entitäten aus der internen Datenhaltung verknüpft.

Die anschließende Aufzählung beschreibt einerseits die notwendigen Use-Cases und die jeweils beteiligten Tabellen aus dem vereinfachten Datenmodell (siehe Abb. 7.5):

- **Projekt anlegen**
  Beteiligte Tabellen:
  - *AriadneProject*
  - *AriadneUser*
- **Use-Case anlegen**
  Beteiligte Tabellen:
  - *AriadneProject*
  - *AriadneUser*
  - *AriadneUseCase*
- **Artefakte anlegen**
  Beteiligte Tabellen:
  - *AriadneUseCase*
  - *AriadneArtefakt*
  - *AriadneUser*

Der Use-Case *Projekt anlegen* setzt voraus, dass ein Benutzer existiert, der entsprechende Rechte besitzt. Deshalb werden an dieser Stelle folgende Use-Cases vorausgesetzt:

- Nutzer anlegen
- Rechte vergeben

Nutzer werden in der Tabelle *AriadneUser* verwaltet. In der Tabelle *AriadnePerspective* werden die Rechte (Perspektiven) für einen Nutzer verwaltet.

40 Das vollständige Datenmodell mit allen Entitäten findet sich in Anhang A.1.

Das Aktivitätsdiagramm (Abb. 7.6) visualisiert einen vereinfachten Ablauf um ein Artefakt in der Baumstruktur (siehe auch Abschnitt 7.2.2) anzulegen bzw. zu ändern.

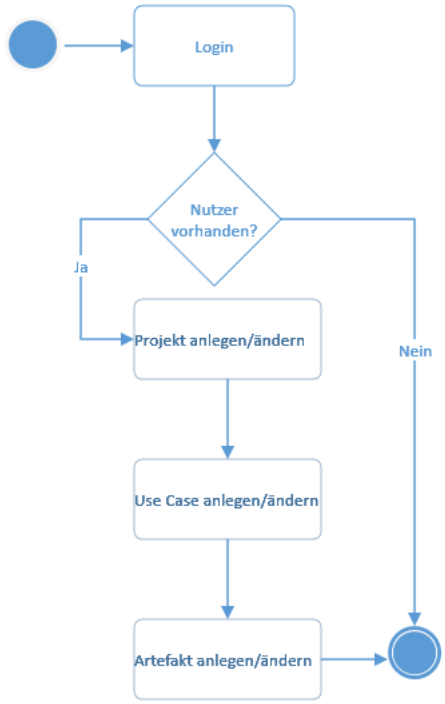


Abbildung 7.6: Aktivitätsdiagramm zum Anlegen eines Artefakts (vereinfachter Ablauf)

7.1.2.2 *AriadneRecommendationDB*

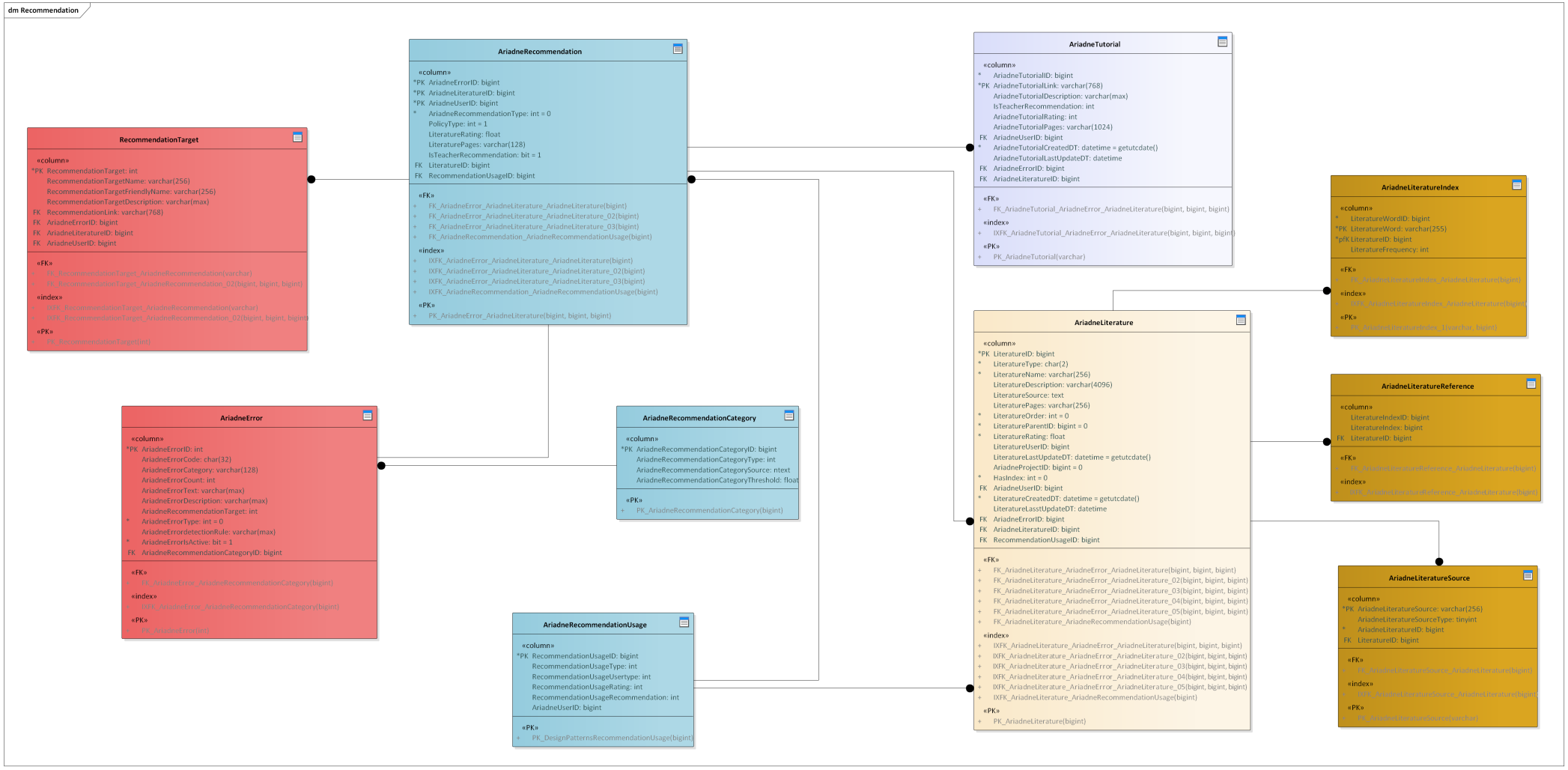


Abbildung 7.7: Vereinfachtes Datenmodell der AriadneRecommendation-Datenbank

Die Datenbank *AriadneRecommendation* ist die zentrale Datenbank zur Verwaltung der Empfehlungen (Abb. 7.7), die Dozierende, Studierenden und Studierende sich untereinander zu Unterlagen und Tutorials geben können (Abb. 7.8). Die Datenbank wurde aufgrund ihrer Größe aus der *Ariadne*-Datenbank ausgelagert. Zudem könnte die *Ariadne*-Datenbank auch dezentral gehostet werden, während dies bei dieser Datenbank aufgrund der Größe ausgeschlossen wurde. Sowohl die *AriadneRecommendation*-Datenbank wie auch die Logging-Datenbank sind außerdem sehr individuell in ihren Inhalten, sodass diese lokal gespeichert werden sollten[41].

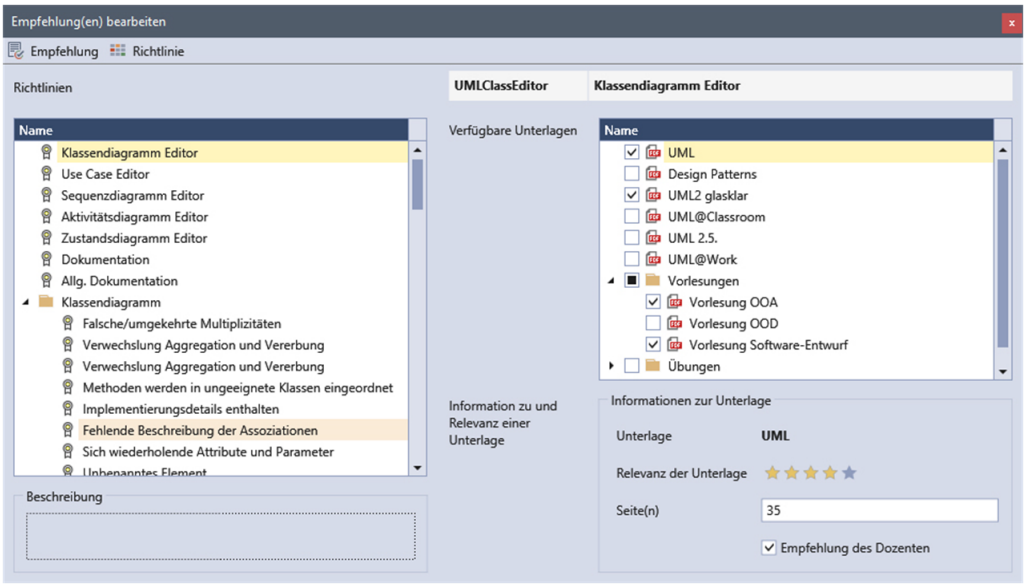


Abbildung 7.8: Dialog, um Empfehlungen für eine Problemstelle zu konfigurieren.

Zentrale Tabellen in dieser Datenbank sind:

- *AriadneLiterature* (Basis für einen Scaffold): Diese Tabelle verwaltet Informationen wie eine ID, den Typ (z. B. PDF oder Office-Dokument), Name, Beschreibung, der Inhalt des Dokuments, das Feld zum Dozententipp, den Ersteller, das vergebene Rating und den Index, mit Hilfe dessen das Dokument durchsucht werden kann und eine Projekt ID, um zu unterscheiden, welche Literatur projektabhängig ist.
- *AriadneTutorials* (Basis für einen Scaffold): Diese Tabelle verwaltet Informationen wie eine ID, URL, Beschreibung, das vergebene Rating, das Feld zum Dozententipp und den Ersteller.

41 Eine vollständiges Datenmodell der *AriadneRecommendation*-Datenbank findet sich in Abb. A.2.

- *AriadneError* (Basis für Empfehlungen): Diese Tabelle verwaltet die in Abschnitt 5.3.5 identifizierten Problemstellen für Diagramme. Es werden Informationen wie Fehler ID, Fehler Code, Fehlertext, Fehlerbeschreibung und die Regel zur Identifikation des Fehlers administriert.
- *AriadneRecommendation* (Basis für Empfehlungen): Mit Hilfe dieser Tabelle wird die eigentliche Empfehlung realisiert. Sie entspricht einer technischen Tabelle und verknüpft Informationen aus *AriadneError*, *AriadneLiterature* und *AriadneTutorials*. Verwaltet werden Informationen wie, Fehler ID, Tutorial ID, Literature ID, User ID, Typ der Empfehlung (Unterlagen oder Tutorials), die Bewertung, empfohlene Seiten und ein Feld zum Dozententipp.

7.1.2.3 *AriadneTelemetry*

Abbildung 7.9 zeigt das Datenmodell der dritten Datenbank, die Ariadne benötigt, um telemetrische Daten von Nutzern erfassen zu können (Abschnitt 7.3.5). Zentrale Tabelle dieser Datenbank ist die Tabelle *log*. Diese Tabelle verwaltet Informationen wie das eingetretene Ereignis, Titel, Zeitstempel und ein Feld *Message*, in dem alle geloggten Informationen als Nachricht im JavaScript Object Notation (JSON)-Format abgelegt werden.

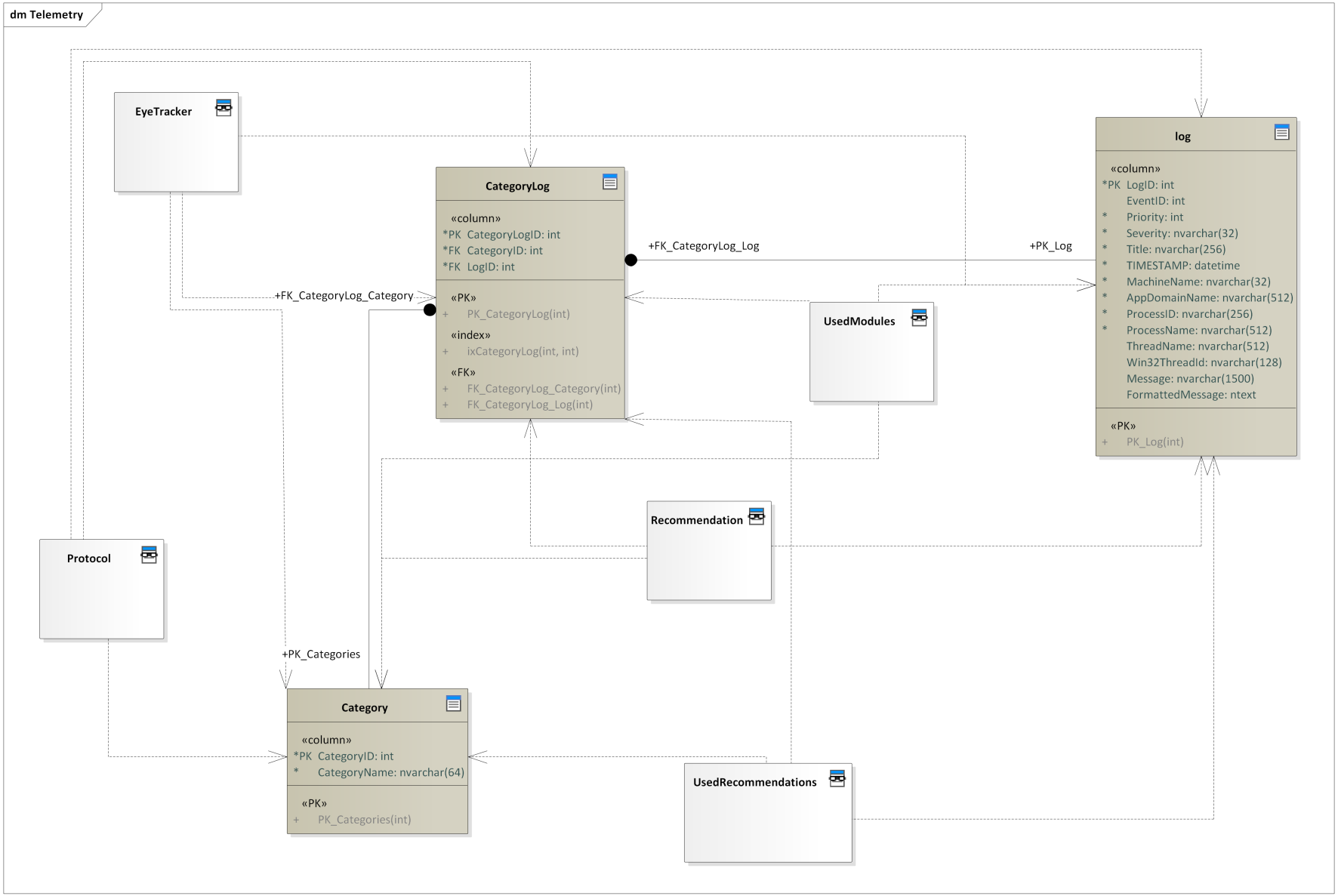


Abbildung 7.9: Datenmodell der *AriadneTelemetry*-Datenbank

7.1.3 *Rollenkonzept*

Das Rollenkonzept von Ariadne entspricht im Wesentlichen den Akteuren, die im Use-Case-Diagramm (Abschnitt 7.1.1.1) zu sehen sind: Es gibt Dozierende in der Rolle *Teacher*, Studierende in der Rolle *Student* und eine weitere Rolle *Administrator*. Die Rechteverwaltung wird in der Datenbank *AriadneDB* geregelt. Dafür sind drei Tabellen vorgesehen: *AriadneUser*, *AriadneUser_AriadnePerspective*, *AriadnePerspective* (siehe Abb. 7.10). In Ariadne wird weniger von Rollen gesprochen, sondern von Perspektiven, da verschiedene Rollen bzw. Nutzer wie in Abschnitt 4.1 beschrieben verschiedene Sichten (Perspektiven) auf ein Softwaresystem einnehmen können.

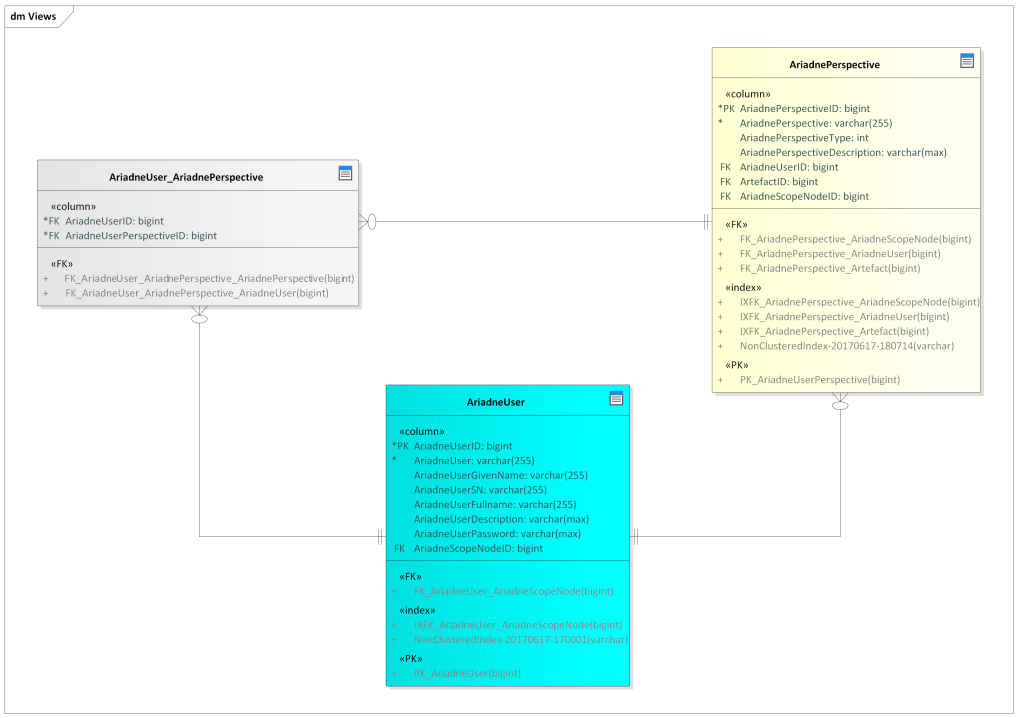


Abbildung 7.10: Datenmodell für die Zuweisung von Rollen in Ariadne

In der Oberfläche von Ariadne können Dozierende und Administratoren, wie in Abb. 7.11 zu sehen ist, Rechte vergeben, die in den entsprechenden Felder verwaltet werden. Diese Rechte betreffen entweder verschiedene Artefakte, aber auch existierende Projekte. Eine Freigabe von Projekten für dedizierte Nutzer ermöglicht somit eine Form der Kollaboration (siehe Abschnitt 6.8).

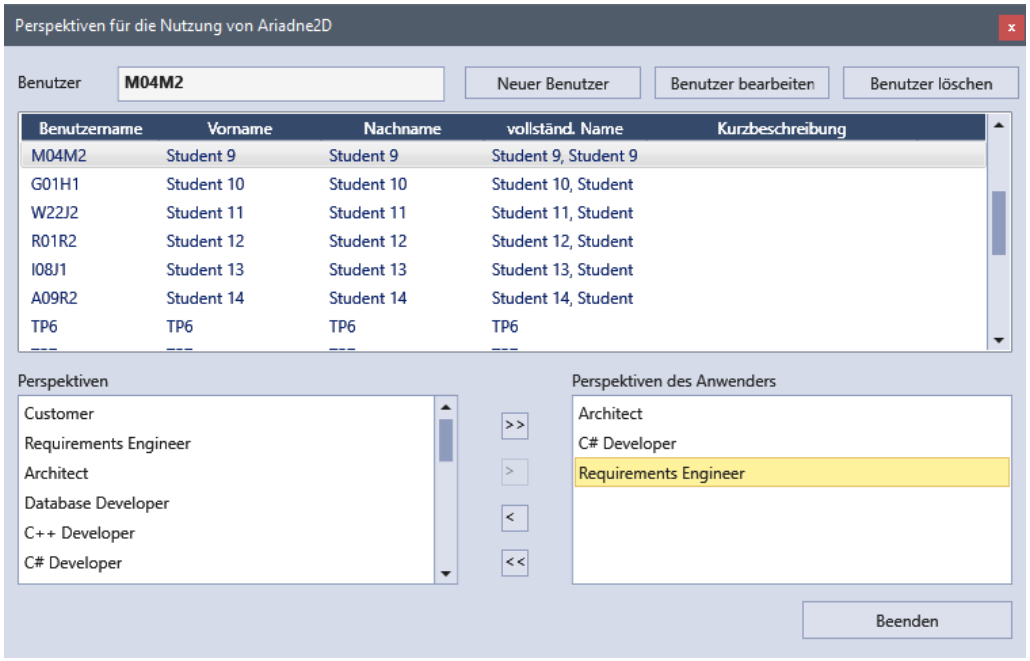


Abbildung 7.11: Zuweisung von Rollen in Ariadne

QC 6 zeigt beispielhaft eine Abfrage der vergebenen Rechte.

```
public static bool HasPerspective(string userName, string
 ↪  perspective){
    bool result = false;
...
        using (var AriadneDB = new DBAccess("AriadneDB")){
            string sqlString = "SELECT dpp.AriadnePerspective"
                     + "     FROM[AriadneUser] dpu"
                     + "    INNER JOIN
                      ↪  AriadneUser_AriadnePerspective dpudpp"
                     + "      ON dpu.AriadneUserID =
                      ↪  dpudpp.AriadnenUserID"
                     + "    INNER JOIN AriadnePerspective dpp"
                     + "      ON dpp.AriadnePerspectiveID =
                      ↪  dpudpp.AriadneUserPerspectiveID"
                     + "    WHERE dpu.AriadneUser = '" + userName +
                      ↪  "' AND dpp.AriadnePerspective like '" +
                      ↪  perspective + "' ";
...
}
```

QC. 6: Abfrage der Perspektiven aus der Datenbank

7.1.4 *Verwendete bestehende Softwarekomponenten*

Bei der Auswahl der bestehenden Softwarelösungen wurde darauf geachtet Medienbrüche zu vermeiden. Da bei der Evaluation von Augmented Reality (AR) die Microsoft Hololens zum Einsatz kam und eine gemeinsame Datenbasis für 2D wie auch AR-Anwendungen angestrebt war, wurden die Werkzeuge entsprechend dieser Anforderungen ausgewählt.

7.1.4.1 *Programmiersprache & Entwicklungsumgebung*

Als Entwicklungsumgebung wurde Visual Studio 2017 Enterprise gewählt. Die Programmiersprache C# wurde gewählt, da eine gemeinsame Datenbasis zwischen einer Windows-Applikation und der AR-Brille bestehen soll.

7.1.4.2 *Windows Presentation Foundation*

Mit Avalon startete Microsoft die Entwicklung einer neuen Bibliothek für grafische Benutzeroberflächen, die 2006 unter dem Namen WPF als Teil von .NET 3.0 veröffentlicht wurde.

Bei der Verwendung von WPF wird die Benutzeroberfläche mit einer an XML angelehnten Sprache beschrieben: mit der in Abschnitt 7.1.1.3 erwähnten XAML. Dadurch wird es möglich, die Beschreibung der Benutzeroberfläche von Quellcode zu trennen (Kühnel, 2013). „WPF-Anwendungen bieten eine umfangreiche Unterstützung von 2D- und 3D-Grafiken. Dabei wird die schnelle Grafikausgabe durch DirectX genutzt. Das wiederum bedeutet, dass die Grafikkarte zur Berechnung der grafischen Elemente herangezogen wird und nicht die CPU. Das führt zu einer deutlich verbesserten Performance“ (Kühnel, 2013). „WPF-Anwendungen bieten vielfältige grafische Unterstützung, z. B. für die Darstellung der Steuerelemente, grafische Animationen, Unterstützung von Videos, Bildern und Audio-Dateien“ (Kühnel, 2013), mit dem Vorteil der vektorbasierten Darstellung.

7.1.4.3 *SQL und SQL-Server*

Für die Datenverwaltung wurden Microsoft-SQL-Datenbanken gewählt und der von Microsoft zur Auswahl gestellte SQL Server Version 17 verwendet. Verwendete Sprachen sind demzufolge Structured Query Language (SQL) und Transact-SQL (TSQL) als prozedurale Erweiterung für Microsoft-SQL-Server. T-SQL ergänzt den SQL-Standard und hat einen minimalen Sprachumfang um Logik zu implementieren. Ergänzt

werden Funktionalitäten wie Prozedurale Programmierung, lokale Variablen, Fehlerbehandlung und Charakteristika zur Zeichenketten- (STRING) Verarbeitung. Zyklische Aufgaben oder Anweisungen, die häufig zum Einsatz kommen, die anderenfalls vom Client (innerhalb des C#-Codes) ausgeführt werden, können als Stored Procedures (gespeicherte Prozeduren) auf dem Datenbankserver gesichert werden. Der Vorteil der Verwendung von T-SQL in der Ariadne-Implementierung ist, dass die Stored Procedures in der Datenbank kompiliert werden und nicht innerhalb der C#-Applikation. Damit gelingt es effizienteren Quellcode (QC) zu erzeugen (Urban, Neumann, Löffelmann & Köller, 2011). *SELECT*-Statements sind in der Regel im C#-Quellcode direkt hinterlegt, da beispielsweise Bilder geladen und damit zur Darstellung konvertiert werden müssen. *UPDATE*-, *INSERT*- und *DELETE*- Statements sind in Stored Procedures ausgelagert.

7.1.4.4 *EnterpriseLibrary*

Die „Microsoft Enterprise Library“ (Version 5.05)[42] ist eine Sammlung von sogenannten Application Blocks aus den Microsoft Patterns und Practices für .Net, die ein einheitliches API für den Zugriff auf Daten, Protokollierung, Caching, uvm. zur Verfügung stellt. Ariadne nutzt vor allem den Logging Application Block. Die Enterprise-Library wird als integrierbare Binärdateien und Quellcode bereitgestellt. Sie ist frei verfügbar. In Ariadne wird damit die Erfassung telemetrischer Daten ermöglicht, d. h. mit Hilfe des Logging Application Blocks können Rohdaten, wie Zeitstempel, Prozess-IDs, Tasknamen etc., im Rahmen der Nutzung von Ariadne, automatisch erfasst und z. B. in einer Datenbank für eine weitere Verarbeitung abgelegt werden. Die strukturierte Ablage solcher Log- bzw. Telemetriedaten erlaubt es Dozierenden weitere Rückschlüsse, z. B. bezogen auf die Verwendung der Scaffolds zu ziehen. Zudem kommt Ariadne hier der einleitend beschriebenen „Türöffner“-Funktion nach. Da beispielsweise auch Bildschirmkoordinaten innerhalb der Applikation erfasst werden können, sind weitere Forschungsarbeiten, beispielsweise im Umfeld von Eyetracking, mit Hilfe der Software möglich. Die Nutzung und Umsetzung der Enterprise Library wird in Abschnitt 7.3.5 weiter ausgeführt.

42 https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff664569(v=pandp.50)

7.1.4.5 *DevExpress*

DevExpress[43], Version 20.02, ist die zentrale Komponente für die Oberfläche von Ariadne. Das Framework bietet Werkzeuge und Komponenten mit spezieller Ausrichtung auf das User Interface.

Die Entwicklung von Ariadne verlief in mehreren Iterationen. Während in dem Experiment, in dem der Einsatz von AR evaluiert wurde, noch eine ältere Version von Ariadne zum Einsatz kam (siehe Abschnitt 6.10.1), wurde mit Hilfe der DevExpress-Bibliothek die Oberfläche von Ariadne noch einmal vollständig überarbeitet. Grund waren die Ergebnisse der Fragebogen- und Think-Aloud-Studie, die wie in den vorherigen Kapiteln erläutert, eine Softwarelösung erfordert, in der sich Nutzer wohl fühlen. Hier wurde das Potenzial von DevExpress gesehen, eine Oberfläche zu schaffen, bei deren Verwendung Nutzer an Office-Produkte denken. Diese Adaption wurde in den Evaluationen (siehe Kapitel 8) mehrmals von Nutzern erwähnt.

7.1.4.6 *OpenNLP*

In Kapitel 6 wurden Scaffolds zur Identifikation von Use-Cases und Klassenkandidaten konzeptuell vorgestellt, die integriert in Ariadne, die Analyse von natürlicher Sprache erfordern. Use-Cases wie Klassenkandidaten und Methoden werden auf Basis von Dokumenten aus natürlicher Sprache abgeleitet. Eine Variante, computergestützt natürliche Sprache zu interpretieren, ist Natural Language Processing (NLP)[44]. Diese Arbeit beschäftigt sich nicht mit der Optimierung oder Verbesserung von Forschungsansätzen zu NLP, nutzt aber bestehende Möglichkeiten aus, Studierenden erste Ansätze zur Analyse von Use-Cases und Klassenkandidaten zu bieten. Einsatzgebiete bzw. Aufgaben und Funktionen von NLP sind laut Jurafsky und Martin (2019) und Ganegedara (2018) unter anderem die Identifikation der Sprache, Tokenisierung, Satzsegmentierung, Part-of-Speech(POS)-Tagging und die Extraktion benannter Entitäten (Ganegedara, 2018, S. 2). Die folgende Aufzählung der wichtigsten Aufgaben von NLP bezieht sich auf Ganegedara (2018):

- Tokenisierung: Bei der Tokenisierung geht es darum, einen Textkorpus in atomare Einheiten (zum Beispiel Wörter) zu zerlegen (Jurafsky & Martin, 2019, S. 15; Ganegedara, 2018, S. 2).
- Wort-Sinn-Disambiguierung (WSD): WSD ist die Aufgabe der Identifizierung der richtigen Bedeutung eines Wortes. WSD wird zur Beantwortung von Fragen eingesetzt (Ganegedara, 2018, S. 3).

43 https://www.devexpress.com/

44 Das Gebiet des NLP ist ein eigener Forschungsbereich

- Name Entity Recognition (NER): NER versucht, aus einem Text Entitäten zu extrahieren (zum Beispiel Personen oder Orte). NER ist ein unerlässliches Thema in Bereichen wie Information Retrieval und Wissensrepräsentation (Ganegedara, 2018, S. 3).
- Part-of-Speech (PoS)-Tagging: Beim POS-Tagging werden Worte ihren Wortarten zugeordnet. Dabei kann es sich um grundlegende Tags wie Substantiv, Verb, Adjektiv, Adverb und Präposition handeln oder aber auch um z. B. Eigennamen (Jurafsky & Martin, 2019, S. 143; Ganegedara, 2018, S. 3).
- Satz/Synopse-Klassifizierung: Die Klassifizierung von Sätzen oder Synopsen (z. B. Filmkritiken) wird durch Training eines Klassifizierungsmodells mit gelabelten Daten, d. h. von Menschen kommentierte Rezensionen, die entweder positiv oder negativ gekennzeichnet sind, erreicht (Ganegedara, 2018, S. 3).
- Sprachgenerierung: Bei der Spracherzeugung wird ein Lernmodell (z. B. neuronales Netz) mit Textkorpora (eine große Sammlung von Textdokumenten) trainiert, die den folgenden neuen Text vorhersagen (Ganegedara, 2018, S. 3).
- Fragenbeantwortung (QA): QA-Techniken besitzen einen hohen kommerziellen Wert und bilden die Grundlage von Chatbots und VA (Virtuellen Assistenten, wie beispielsweise Apple Siri). Inzwischen werden Chatbots von vielen Unternehmen für den Kundensupport eingesetzt (Ganegedara, 2018, S. 3).

In Ariadne kommen die Satzerkennung und die Tokenisierung der Apache OpenNLP Bibliothek zum Einsatz, eine kostenfreie Implementierung für NLP[45]:

- Satzerkennung: Der „SentenceDetector“ erkennt, ob ein Punkt das Ende eines Satzes markiert oder ob er eine andere Bedeutung hat. Die Angabe eines trainierten Modells ist notwendig. OpenNLP liefert Modelle für verschiedenen Sprachen, z. B. „de-sent.bin“ für Satzerkennung in deutschen Texten[46].
- Tokenisierung: Bei der Tokenisierung werden Zeichenfolgen in Token zerlegt. Token sind beispielsweise Worte, Satzzeichen oder Zahlen etc.[47].
- Extraktion benannter Entitäten: Der „TokenNameFinder“ kann benannte Objekte und Zahlen im Text erkennen. Um Entitäten bemerken zu können, wird ein Modell benötigt. Das Modell ist abhängig

45 https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html
46 http://opennlp.sourceforge.net/models-1.5/
47 https://stanbol.apache.org/docs/trunk/components/enhancer/engines/opennlppos.html

von der Sprache und dem Entitätstyp, für den es trainiert worden ist. Das OpenNLP-Projekt bietet vortrainierte Modelle, die auf verschiedenen frei verfügbaren Corpora trainiert worden sind.

Jede dieser Komponenten ist über eine API zugänglich. Es werden vortrainierte Modelle verwendet. Die API kann über ein NuGet-Paket geladen werden.

7.2 ARIADNE

Die folgenden Abschnitte führen die zentralen Komponenten von Ariadne ein: Zunächst wird die Oberfläche kurz vorgestellt, anschließend die wichtigste Navigationsstruktur erläutert und die Umsetzung der Diagrammtypen, sowie die Integration der Scaffolds beschrieben.

7.2.1 *Benutzerschnittstelle*

Die Oberfläche von Ariadne wurde unter Verwendung der UI-Komponenten der devexpress Bibliothek (Abschnitt 7.1.4.5) designed.

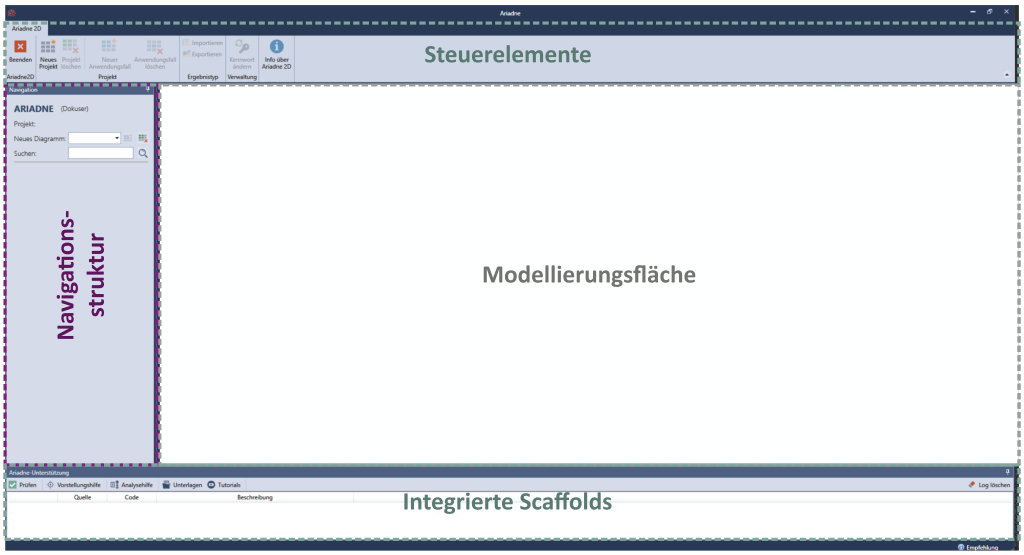


Abbildung 7.12: Oberfläche von Ariadne

Abbildung 7.12 zeigt die vier wesentlichen Elemente:

- Die Steuerungselemente sind in der Oberfläche (Grafical User Interface (grafische Benutzeroberfläche) (GUI)) oben angeordnet.
- Auf der linken Seite der GUI, befindet sich die zentrale Baumstruktur, in der zwischen Artefakten navigiert werden kann.

- Mittig-Rechts befindet sich die Modellierungsfläche und die Toolbox der Editoren.
- Die integrierten Scaffolds sind unten angeordnet. Diese sind standardmäßig nicht modale Dialoge, d. h. sie können parallel zur Bearbeitung geöffnet werden.
- Die Umgebung der Scaffolds, der Steuerungselemente und der Navigationsstruktur können im Layout verändert werden und besitzen einen „Anheftstatus".

7.2.2 *Navigationsstrukturen*

Wie in Abschnitt 6.2 vorgestellt, ist die Navigationsstruktur innerhalb eines Projektes eingeschränkt und prinzipiell nach dem Vorschlag aus Lehrbüchern, wie z. B. Balzert und Balzert (2009), Rupp und die SOPHISTen (2014) ausgearbeitet, sodass Zusammenhänge zwischen einzelnen Diagrammen und der vorgelagerten Anforderungsanalyse deutlich werden. Wie im Problemkatalog schon ausgeführt (Abschnitt 5.3.5, z. B. Tabelle 5.27 oder Tabelle 5.31), wurden diesbezüglich Schwierigkeiten bei Studierenden festgestellt.

Das Öffnen und Erzeugen neuer Diagrammtypen und damit Editoren ist nur in der vorgegebenen Struktur (siehe Abb. 7.13) möglich:

- Für ein Projekt sind standardmäßig ein Use-Case-Diagramm und eine Projektdokumentation vorgesehen.
- Unterhalb eines Projektes können einzelne Use-Cases für das Projekt angelegt werden.
- Ein Use-Case wird dazu in der Navigationsstruktur erstellt. Er enthält eine textuelle Beschreibung, eine strukturierte Beschreibung (Abschnitt 4.4) sowie ein Klassendiagramm. Dadurch wird im Wesentlichen der Einsatz der OOA-Methode (Schritt 1-3; siehe Abschnitt 4.3) ermöglicht. Bei Bedarf kann ein Aktivititätsdiagramm hinzugefügt werden oder weitere untergeordnete Use-Cases.
- Unterhalb des Klassendiagramms können beliebig viele weitere Diagrammtypen (Klassendiagramme, Aktivitätsdiagramme, Zustandsautomaten, Sequenzdiagramme) angelegt werden (auch weitere Use-Cases sind möglich, da möglicherweise Aspekte eines geplanten Use-Cases tiefergehend betrachtet werden müssen). Hier werden die Schritte vier und fünf, die parallele Erstellung von statischem und dynamischen Modell, ermöglicht.

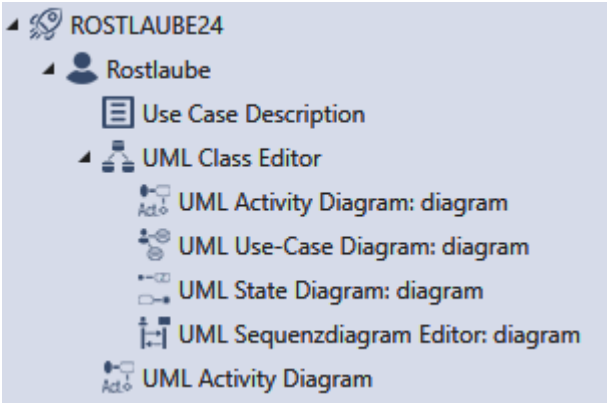


Abbildung 7.13: Beispiel für die Baumstruktur

Die Navigationsstruktur ist über diese Regeln hinaus vollständig dynamisch und abhängig von Nutzerentscheidungen. Sie ist als Baumstruktur organisiert um Studierenden die Zusammenhänge zwischen Artefakten zu verdeutlichen. Zentrale Methoden, die für die Anzeige der Baumstruktur eine Rolle spielen sind beispielhaft in QC 7 für das Hinzufügen eines neuen Use-Cases dargestellt.

```
public void AddUseCase() {
...
    var addUseCaseDialog = new AddUseCaseDialog();
    addUseCaseDialog.UserID = UserID;
     // find the selected project name
    addUseCaseDialog.ProjectName = _mainWindow.ProjectName;
...
     if (addUseCaseDialog.ShowDialog() == true){
        _mainWindow.ShowWait();
         // only for a valid UseCaseDialogName
            if (!string.IsNullOrEmpty(addUseCaseDialog.UseCase)){
               UseCaseID = addUseCaseDialog.UseCaseID;
               UseCase = addUseCaseDialog.UseCase;
               AddScopeNodeUseCase( projectName,
                ↪  addUseCaseDialog.UseCase);
               // rebuild the tree for all use cases
              UseCase = string.Empty;
              RefreshScopeAsync();
            }
         _mainWindow.HideWait();
...
```

QC. 7: Use-Cases hinzufügen

Das in Abb. 7.14 dargestellte Sequenzdiagramm zu QC 7 visualisiert den internen Ablauf, der das Hinzufügen eines Use-Cases realisiert.

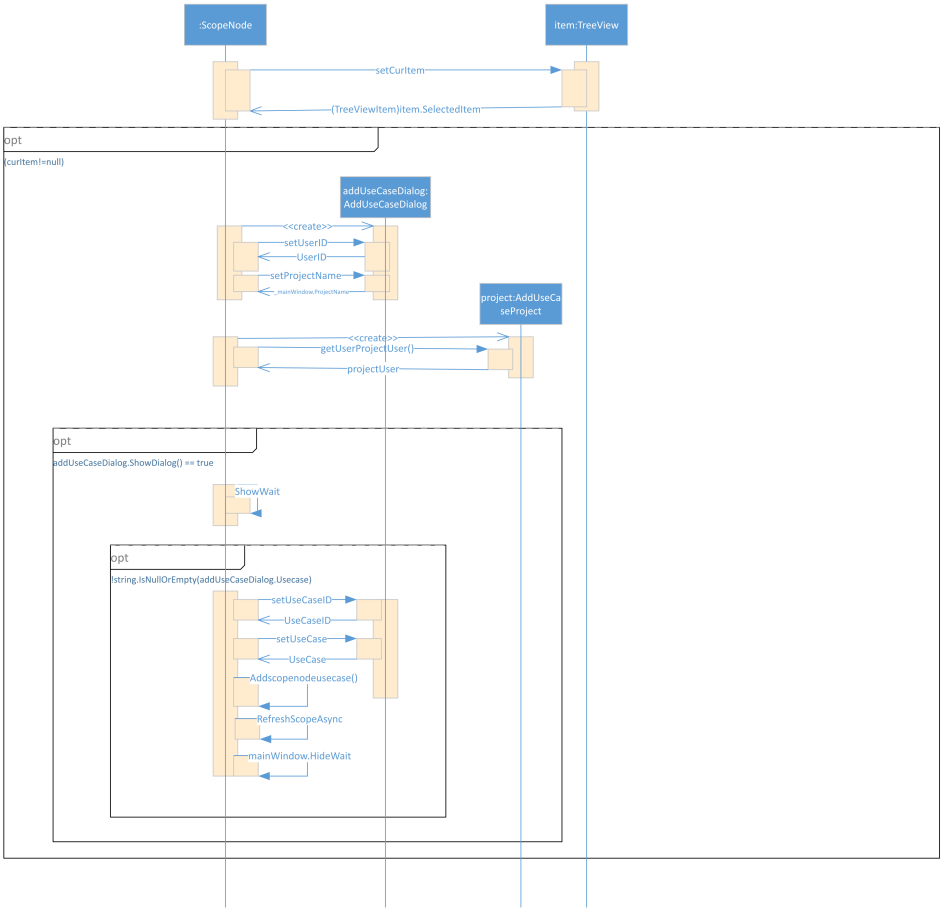


Abbildung 7.14: Sequenzdiagramm für das Hinzufügen eines Use-Cases

Abbildung 7.15 zeigt exemplarisch den Dialog, wie ein Use-Case-Diagramm mittels GUI angelegt wird. Alle anderen Diagramme werden analog angelegt.

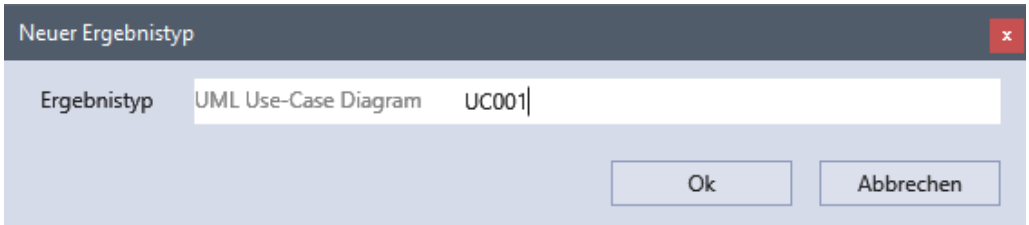


Abbildung 7.15: Dialog zum Anlegen eines Use-Case-Diagramms

7.2.3 *Text- und Modelleditoren*

In Ariadne werden zwei Arten von Editoren zur Verfügung gestellt. Zum einen gibt es einen Texteditor, der grundlegende Textverarbeitung anbietet und zum anderen die Editoren zur Modellierung von Use-Case-Diagrammen, Klassendiagrammen, Aktivitätsdiagrammen, Zustandsdiagrammen und Sequenzdiagrammen (Abschnitt 6.2).

7.2.3.1 *Texteditoren*

Die objektorientierte Analyse beginnt damit, Use-Cases auf Basis von Anforderungsdokumenten zu identifizieren, d. h. eine Dokumentation von Anforderungen geht im Idealfall voraus, bevor die objektorientierte Analyse eines Softwaresystems beginnen kann. Diese Aufgabe wird in Ariadne durch die Bereitstellung eines eigenen Texteditors unterstützt.

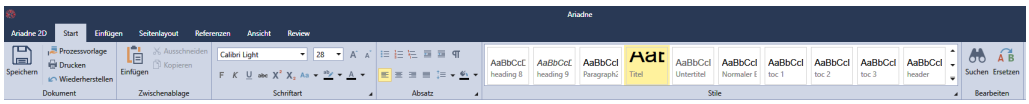


Abbildung 7.16: Steuerelemente des Texteditors

Abbildung 7.16 zeigt die grundlegenden Funktionen des Editors, daneben gibt es beispielsweise noch eine Kommentarfunktion. Besonders anzumerken ist hier die Möglichkeit bestehende Prozessvorlagen als Templates (Abb. 7.17) zu laden. Dozierende können weitere Templates im Bereich der Unterlagen hinterlegen, die dann für Studierende unter dem Button „Prozessvorlagen“ geladen werden können (siehe Abschnitt 6.3.1). Die Arbeit mit Templates wird von einigen Autoren auch als Scaffold betrachtet (siehe (Hislop & Ellis, 2009; Linn, 1992)). Abbildung 7.17 zeigt die GUI, mit deren Hilfe Studierende eine Prozessvorlage in den Texteditor laden können. Beispielhaft sind Prozessvorlagen aus dem Rational Unified Process (Kruchten, 2004) hinzugefügt worden, die für die Textverarbeitung angeboten werden.

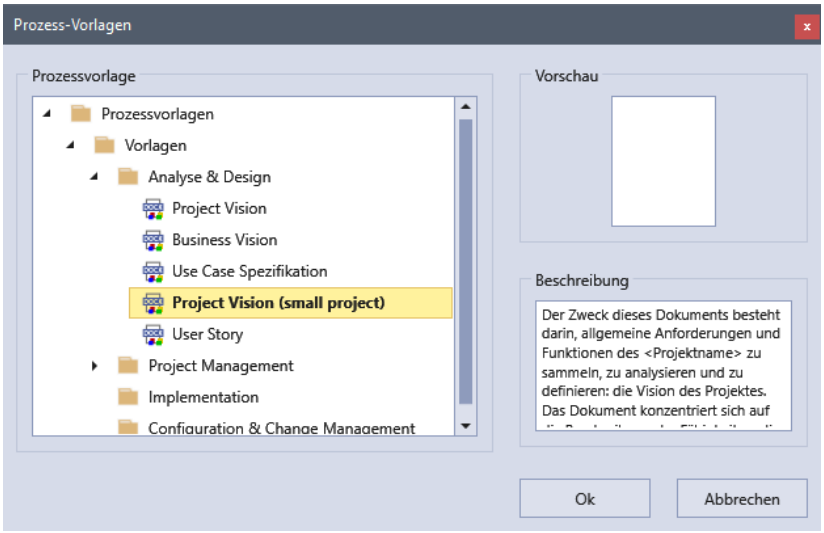


Abbildung 7.17: Prozessvorlagen

7.2.3.2 *Modelleditoren*

Insgesamt gibt es fünf Editoren zur Modellierung des jeweiligen UML-Diagrammtyps. In Abschnitt 6.2 wurden Entscheidungen zur Auswahl und zum Design der Editoren bereits erläutert. Die Editoren bieten bewusst eine eingeschränkte Toolbox an und erzeugen Syntax nicht automatisiert um Studierenden einerseits komplexe Prozesse in der Bedienung eines Editors abzunehmen und andererseits vor allem den Prozess im Lernen der UML-Syntax anzuregen. Ziel ist es, Studierende nicht mit einer überladenen Toolbox zu überfordern, aber dennoch durch die nicht vorhandene Vervollständigung von Syntax Studierende zur selbstständigen Einhaltung dieser aufzufordern. Jeder Editor enthält eine eigene Toolbox, sodass das Problem, dass Elemente ausgewählt werden, die laut UML-Metamodell nicht für den Diagrammtyp vorgesehen sind, nicht auftreten kann.

Im Allgemeinen ist jedes Diagramm aus inhaltlichen Objekten (Content) und Konnektoren (Connectors) aufgebaut. Konnektoren haben insbesondere folgende Eigenschaften:

- *CanDragBeginPoint*: bestimmt die Möglichkeit einen Connector mit dem Beginn an ein Element zu binden
- *CanDragEndPoint*: bestimmt die Möglichkeit einen Connector mit dem Ende an ein Element zu binden
- *CustomStyleId*: beschreibt den internen Identifier
- *CanEdit*: bestimmt Editierbarkeit
- *EndArrow*: beschreibt Typ der Pfeilspitze am Ende des Connectors

- *BeginArrow*: beschreibt Typ der Pfeilspitze am Beginn des Connectors
- *ToolTip*: definiert ToolTip des Connectors

Tabelle 7.1 zeigt die möglichen Konnektoren.

Tabelle 7.1: Mögliche Konnektoren für die Diagrammtypen

CustomStyleId	UML-Bedeutung
Inheritance	Vererbungsbeziehung im Klassendiagramm
Aggregation	Aggregationsbeziehung im Klassendiagramm
Association	Assoziationsbeziehung im Klassendiagramm
directed Association	gerichtete Assoziationsbeziehung im Klassendiagramm
Composition	Kompositionsbeziehung im Klassendiagramm
Realization	Realisierungsbeziehung im Klassendiagramm
Dependency	Abhängigkeitsbeziehung im Klassendiagramm
Inheritance	Vererbungsbeziehung im Use-Case-Diagramm
Extend	Extends-Beziehung im Use-Case-Diagramm
Association	Assoziationsbeziehung im Klassendiagramm
Includes	Includes-Beziehung im Use-Case-Diagramm
Connect	Verbinder im Aktivitätsdiagramm
responseMessage	Antwortnachrichten im Sequenzdiagramm
asyncMessage	Asynchrone Nachrichten im Sequenzdiagramm
message	Synchrone Nachrichten im Sequenzdiagramm
messageToSelf	Selbstaufruf im Sequenzdiagramm

CustomStyleId	UML-Bedeutung
objectConstruction	Nachricht zur Objekterzeugung (*Create*) im Sequenzdiagramm
lifeLine	Lebenslinie im Sequenzdiagramm
Connect	Verbinder im Zustandsdiagramm

Nachfolgend wird nun beispielhaft vorgestellt, wie eine Toolbox in Ariadne realisiert wurde. QC 8 beschreibt die Darstellung der Vererbungsbeziehung im Klassendiagramm-Editor. QC 9 beschreibt die Darstellung der eigentlichen Klasse im Klassendiagramm-Editor.

```
static void InitializePredefinedStencil(DiagramStencil stencil)
...
    stencil.RegisterTool(new FactoryItemTool(
        "Vererbung",
        () => "Vererbung",
        diagram => new DiagramConnectorEx()        {
            CanDragBeginPoint = true,
            CanDragEndPoint = true,
            CanEdit = true,
            CustomStyleId = "Inheritance",
            EndArrowSize = new Size(12, 12),
            EndArrow = ArrowDescriptions.ClosedASMEArrow,
            ToolTip = "Vererbung"
        },
        new Size(120, 80), true));
...
```

QC. 8: Elemente in der Toolbox des Klassendiagramm-Editors

```
...
stencil.RegisterTool(new FactoryItemTool(
                "Klasse",
                () => "Klasse",
                diagram =>
                {
                    AriadneClass model = new AriadneClass();
                    return new DiagramContentItemEx()
                    {
                        CanAttachConnectorBeginPoint = true,
                        CanAttachConnectorEndPoint = true,
                        CanResize = true,
                        CanSnapToThisItem = true,
                        CustomStyleId = "classContentItem",
                        Foreground = Brushes.White,
                        Content = model
                    };
                }));
...
```

QC. 9: Darstellung der Klassenobjekts im Klassendiagramm-Editor

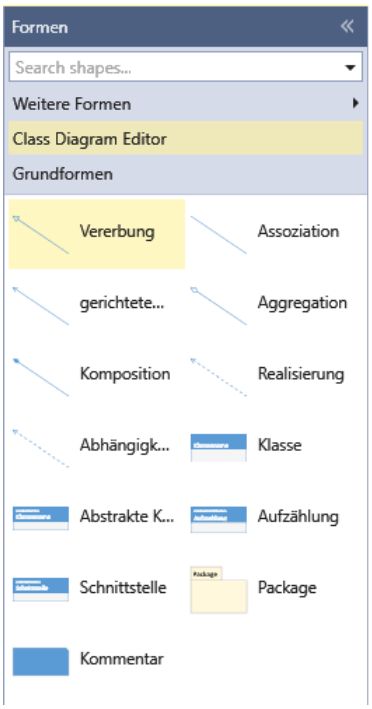


Abbildung 7.18: Toolbox für den Klassendiagramm-Editor

Abbildung 7.18 zeigt exemplarisch die Toolbox für den Klassendiagramm-Editor. Für alle anderen Editoren ist die Funktionsweise analog implementiert.

7.2.4 *Diagramm- und Problemspezifische Scaffolds*

Insgesamt wurden in Ariadne die in Kapitel 6 vorgestellten Scaffolds (abgesehen von AR, das noch weiter evaluiert werden muss und in einer 2D-Umgebung ohnehin nicht integrierbar ist) integriert (siehe auch Abb. 7.19):

- Vorstellungshilfen
  - EMMEs
  - Stakeholder des Projekts
- Analysehilfen (Abhängig vom aktiven Artefakt):
  - Identifikation von relevanten Textstellen für Beschreibungen von Use-Cases
  - Identifikation von Use-Cases
  - Identifikation von Klassenkandidaten
- Unterlagen
  - Allgemeine Literatur
  - Projektspezifische Literatur
  - Beispiele
  - Vorlagen
- Tutorials
  - Online Material

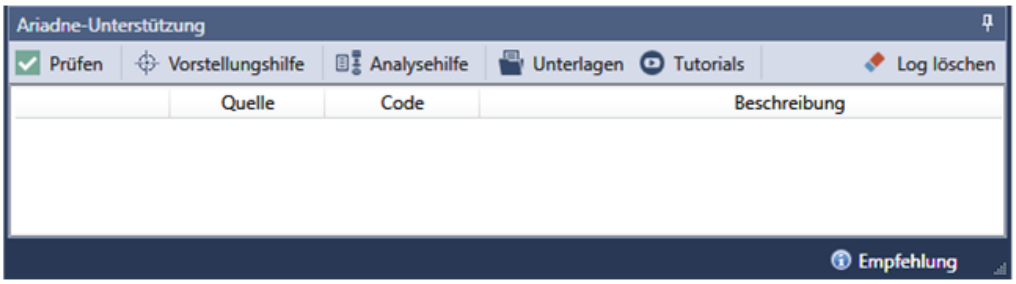


Abbildung 7.19: Integrierte Scaffolds in Ariadne

Der folgende Abschnitt legt besonderes Augenmerk auf die Realisierung der Anforderung Studierenden Scaffolds diagrammspezifisch und problemspezifisch zu empfehlen:

1) Abhängig des gerade aktiven Editors, ist eine bestimmte Auswahl bzw. Nutzung an Scaffolds möglich:

- Bei der Aktivierung der Projektdokumentation ist die Nutzung der Use-Case-Analyse möglich (Abschnitt 6.9.1).
- Bei der Bearbeitung des Use-Case-Template ist die Nutzung der Substantiv-Verb-Analyse zur Identifikation von Klassenkandidaten möglich (Abschnitt 6.9.2).
- Das EMME kann abgespielt werden, wenn ein Sequenzdiagramm-Editor aktiviert wird.

Die Nutzung der Scaffolds ist nicht gleichbedeutend damit, dass der Scaffold nicht in einer anderen Ansicht geöffnet werden kann, es kann allerdings keine Analyse mehr durchgeführt werden.

2) Die Scaffolds Unterlagen und Tutorials können problemspezifisch empfohlen werden.

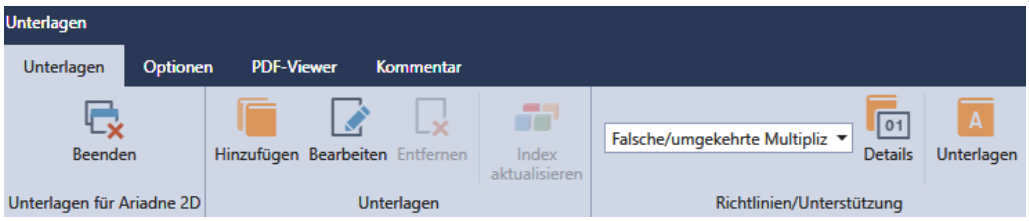


Abbildung 7.20: Empfehlungen bearbeiten

Dazu verfügt ein Nutzer in der Rolle des Dozierenden über die Zusatzfunktion, konkrete Empfehlungen für ein Problem (siehe Abb. 7.20, konkret unter Reiter *Richtlinien/Unterstützung* und dem Button *Unterlagen*) zu konfigurieren.

- Ein 5-Sterne-Rating, das gesammelt über alle Scaffolds priorisiert angezeigt wird
- Seitenzahlen, die, wenn sie definiert sind, dabei helfen, möglichst effizient, Hilfestellungen zu dem Problem direkt zu erhalten
- Eine Checkbox, mit Hilfe dieser Dozierende besonders relevante Unterlagen markieren können

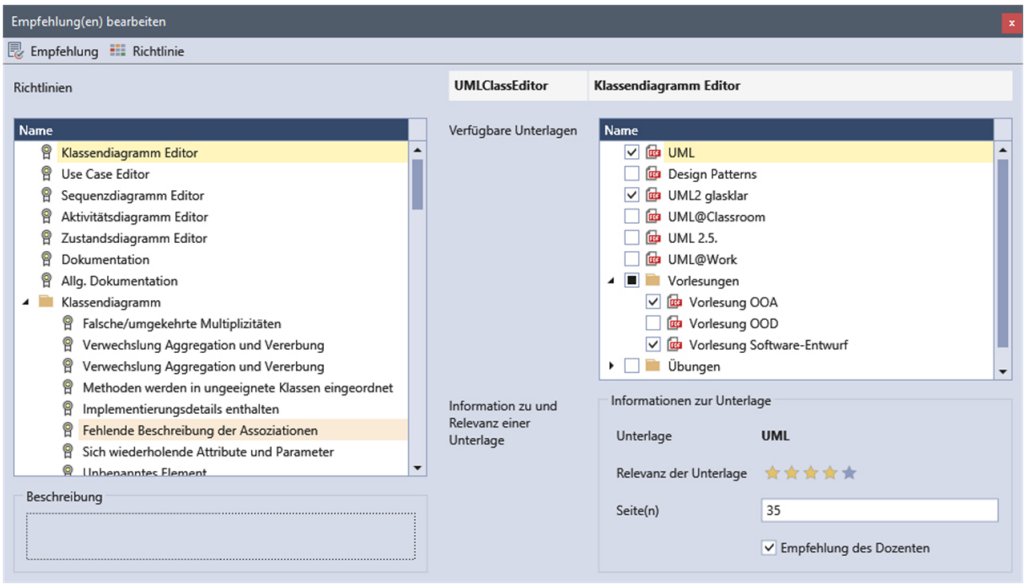


Abbildung 7.21: Grafische Oberfläche zum Hinzufügen oder Bearbeiten von Empfehlungen im Scaffold Unterlagen

Über diese Funktion gelangt der Nutzer zu einem Dialog, der es ermöglicht problemspezifisch Empfehlungen abzugeben. Dazu wählt der Dozierende das Problem im Dropdown-Menü in der Auswahl Richtlinien aus und kann anschließend in der rechten Auswahlbox Unterlagen markieren, die für dieses Problem eine Hilfestellung enthalten (Abb. 7.21). Unterhalb der Box können noch weitere Attribute für die gewählten Unterlagen definiert werden:

Ergebnisse Meldungen

	AriadneRecommendationID	AriadneLiteratureID	AriadneUserID	AriadneRecommendationType	PolicyType	LiteratureRating	LiteraturePages	IsTeacherRecommendation
1	1	6	1	1	1	4	7; 29; 32	1
2	1	10040	1	1	1	3	2;40	0
3	1	10042	1	1	1	2	66	0
4	3	6	1	1	1	4	12; 17; 29	1
5	3	20	1	1	1	4	2;4	1
6	3	22	1	0	1	5	1; 89	1
7	3	26	1	0	1	2	3;89	1
8	20	6	1	1	1	4	7; 29; 32	0
9	20	10040	1	0	1	3	2;40	0
10	20	10042	1	0	1	2	66	0
11	1212	6	1	0	1	0		1
12	1212	20	1	0	1	3	7;45	1
13	1213	6	1	0	1	1		1
14	1213	21	1	0	1	5	15;23	1
15	1215	6	1	0	1	4		1
16	1215	21	1	0	1	4		1
17	1215	25	1	0	1	0		1
18	1221	6	1	0	1	5	124;244	1
19	1221	16	1	0	1	5	124;244	1

Abbildung 7.22: Auszug aus der Datenbanktabelle, die zu einem Problem spezifische Literatur speichert.

Abbildung 7.22 zeigt einen Auszug aus der Datenbanktabelle, die zu einem Problem (*ErrorID*) Unterlagen und Attribute speichert (z. B. *IsTeacherRecommendation*, *LiteraturePages*, etc.)

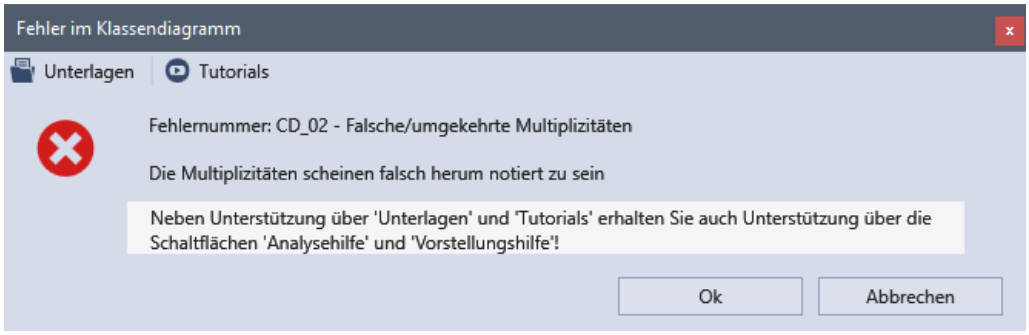


Abbildung 7.23: Dialog zur Anzeige des detektierten Problems

Die Funktionsweise für Empfehlungen von Tutorials ist ähnlich, Dozierende können hier allerdings keine Seitenzahlen empfehlen. Über diese Funktion ist gewährleistet, dass Studierende, wenn sie ein Diagramm prüfen lassen und entsprechende Empfehlungen hinterlegt sind, problemspezifisch Empfehlungen erhalten. Dies geschieht über den Dialog, der nach dem Klick auf ein detektiertes Problem angezeigt wird (Abb. 7.23).

Sollten Empfehlungen für einen Diagrammtyp hinterlegt worden sein, werden diese, ebenfalls in Unterlagen und Tutorials, diagrammspezifisch angezeigt (Abb. 7.24).

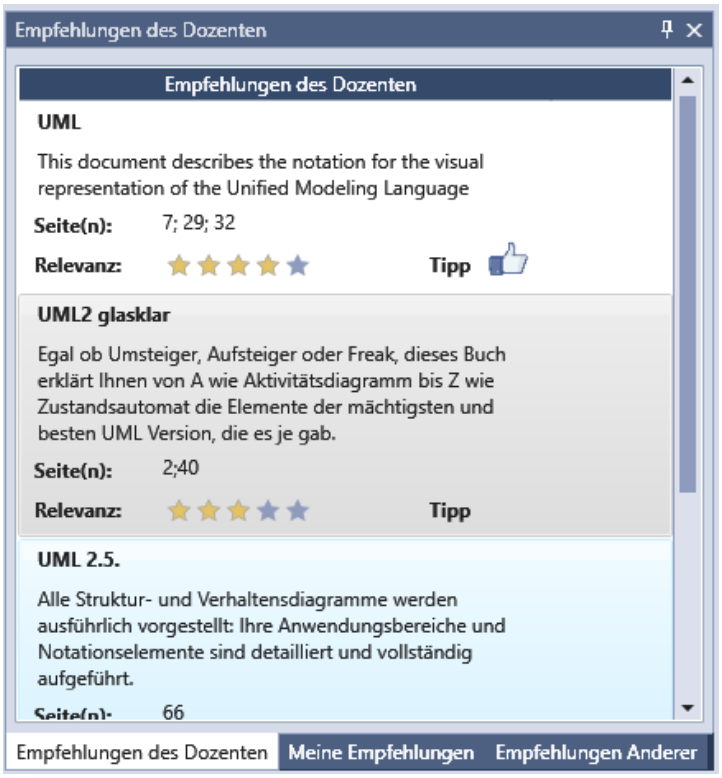


Abbildung 7.24: Anzeige der empfohlenen Scaffolds

7.3 AUSGEWÄHLTE IMPLEMENTIERUNGSKONZEPTE

In diesem Kapitel werden ausgewählte Implementierungskonzepte vorgestellt, die in Ariadne realisiert wurden.

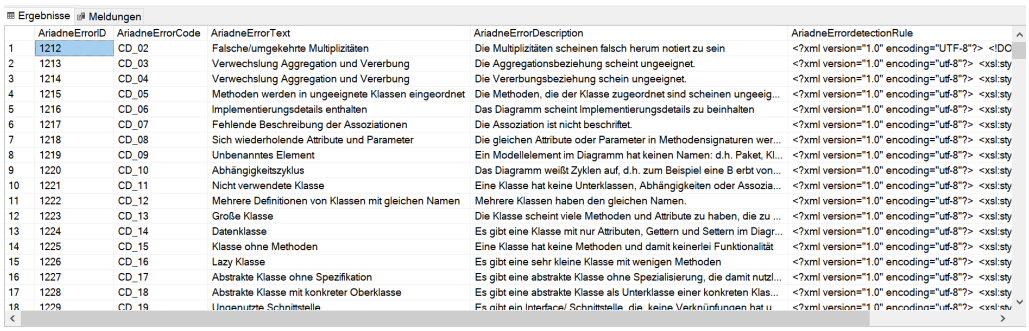
Folgende Konzepte sollen dabei betrachtet werden:

- die Erstellung des Zusammenhangs zwischen einem erkannten Problem und einem Scaffold
- die Funktionsweise der Bewertung bei den Scaffolds *Unterlagen und Tutorials*
- die Funktionsweise zur Detektion der Probleme, die in einem Diagramm erkannt werden können
- die Funktionsweise der Textanalyse
- die Möglichkeit zur Erfassung von nutzerspezifischen telemetrischen Daten

7.3.1 *Problemdefinition und -detektion*

Ariadne bietet Studierenden die Möglichkeit das erstellte Diagramm *Prüfen* zu lassen. Nach der Prüfung werden alle Problemstellen, analog zum

Aufbau von Entwicklungsumgebungen zum Programmieren, unterhalb der Scaffolds gelistet. Nach Auswahl einer Problemstelle werden die betreffenden Elemente rot markiert und ein Dialog mit der Beschreibung angezeigt. Studierende gelangen dann, wenn entsprechende Empfehlungen für das identifizierte Problem hinterlegt sind (siehe Abb. 7.25), über den Button Unterlagen oder Tutorials im Dialog zu problemspezifischer Hilfe.



Ergebnisse | Meldungen

	AriadneErrorID	AriadneErrorCode	AriadneErrorText	AriadneErrorDescription	AriadneErrordetectionRule
1	1212	CD_02	Falsche/umgekehrte Multiplizitäten	Die Multiplizitäten scheinen falsch herum notiert zu sein	<?xml version="1.0" encoding="UTF-8"?> <IDC
2	1213	CD_03	Verwechslung Aggregation und Vererbung	Die Aggregationsbeziehung scheint ungeeignet.	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
3	1214	CD_04	Verwechslung Aggregation und Vererbung	Die Vererbungsbeziehung schein ungeeignet.	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
4	1215	CD_05	Methoden werden in ungeeignete Klassen eingeordnet	Die Methoden, die der Klasse zugeordnet sind scheinen ungeeig...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
5	1216	CD_06	Implementierungsdetails enthalten	Das Diagramm scheint Implementierungsdetails zu beinhalten	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
6	1217	CD_07	Fehlende Beschreibung der Assoziationen	Die Assoziation ist nicht beschriftet.	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
7	1218	CD_08	Sich wiederholende Attribute und Parameter	Die gleichen Attribute oder Parameter in Methodensignaturen wer...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
8	1219	CD_09	Unbenanntes Element	Ein Modellelement im Diagramm hat keinen Namen: d.h. Paket, Kl...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
9	1220	CD_10	Abhängigkeitszyklus	Das Diagramm weißt Zyklen auf, d.h. zum Beispiel eine B erbt von...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
10	1221	CD_11	Nicht verwendete Klasse	Eine Klasse hat keine Unterklassen, Abhängigkeiten oder Assozia...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
11	1222	CD_12	Mehrere Definitionen von Klassen mit gleichen Namen	Mehrere Klassen haben den gleichen Namen.	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
12	1223	CD_13	Große Klasse	Die Klasse scheint viele Methoden und Attribute zu haben, die zu ...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
13	1224	CD_14	Datenklasse	Es gibt eine Klasse mit nur Attributen, Gettern und Settern im Diagr...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
14	1225	CD_15	Klasse ohne Methoden	Eine Klasse hat keine Methoden und damit keinerlei Funktionalität	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
15	1226	CD_16	Lazy Klasse	Es gibt eine sehr kleine Klasse mit wenigen Methoden	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
16	1227	CD_17	Abstrakte Klasse ohne Spezifikation	Es gibt eine abstrakte Klasse ohne Spezialisierung, die damit nutzl...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
17	1228	CD_18	Abstrakte Klasse mit konkreter Oberklasse	Es gibt eine abstrakte Klasse als Unterklasse einer konkreten Klas...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty
18	1229	CD_19	Ungenutzte Schnittstelle	Es gibt ein Interface/ Schnittstelle, die keine Verknüpfungen hat u...	<?xml version="1.0" encoding="utf-8"?> <xsl:sty

Abbildung 7.25: Datenbankauszug aus der Tabelle, die Problembeschreibungen verwaltet

Abb. 7.25 zeigt beispielhaft die tabellarische Aufstellung der erfassten Problemstellen. Es gibt eindeutige IDs, einen diagrammabhängigen Code, der auch dem Nutzer angezeigt wird, den Fehlertext, eine Beschreibung und die zugehörige Regel. Wie bereits in Abschnitt 6.5 beschrieben, ist die Identifikation der Probleme in Diagrammen subjektiv (siehe Ali und Terengganu (2007), Cheers et al. (2019), Gelhausen et al. (2008) und Reischmann und Kuchen (2019)) und kann in der vorliegenden Arbeit nicht vollständig erfasst werden. Deshalb bietet Ariadne Dozierenden die Möglichkeit, wie in Abb. 7.26 bis 7.27 gezeigt, selbstständig Richtlinien zu ändern und hinzuzufügen.

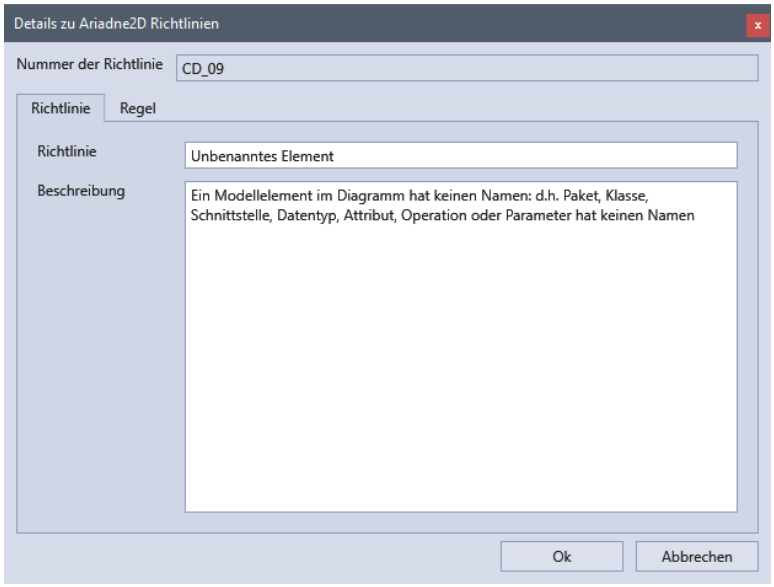


Abbildung 7.26: Beispiel für die Benennung einer Richtlinie

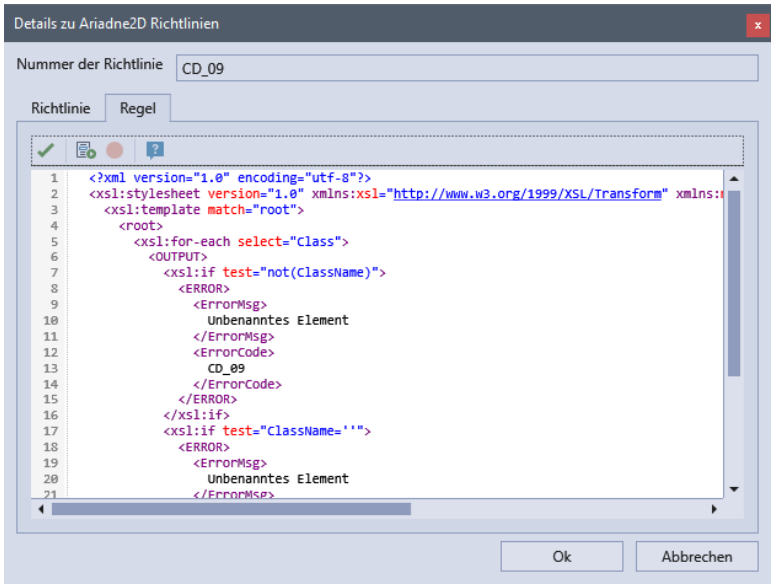


Abbildung 7.27: Beispiel für die Beschreibung einer Richtlinie

Die Validierung eines Diagramms folgt dem Schema, das in Abb. 7.28 dargestellt ist. Alle Diagramme werden in einer XML-Struktur in der Ariadne-Datenbank abgelegt. Dafür wurde eine eigene XML-Struktur konzipiert bzw. die Struktur von devexpress erweitert. Zur Detektion der Problemstellen wird mittels XSLT eine Transformation des XMLs vorgenommen und so ein XML erstellt, das um die gefunden Problemstellen angereichert wird. Es gibt für jede Problemstelle ein XSLT, das zu

dem entsprechenden *Errorcode* in der Tabelle *AriadneError* der *Ariadne-Recommendation*-Datenbank hinterlegt ist (siehe Abb. 7.25). Ein *Errorcode* besitzt eine Kennung, anhand dessen identifiziert werden kann welcher Diagrammtyp betroffen ist. Das XSLT wird auf die XML Daten angewandt und Regelverstöße, wie in XSLT für den Diagrammtyp definiert, werden detektiert. Das Ergebnis der Überprüfung wird wieder an den Diagrammeditor übergeben und fehlerhafte Stellen im Diagramm werden rot markiert.

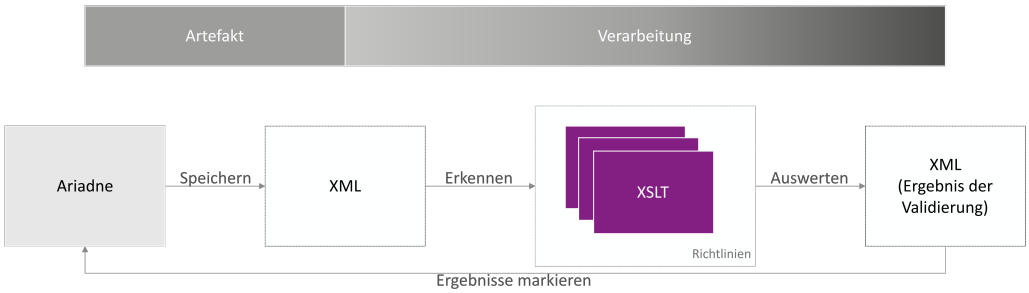


Abbildung 7.28: Ablauf der Problemerkennung in einem Diagramm

Das Beispiel in Abb. 7.29 zeigt einerseits die detektierten Probleme, diese werden unterhalb der Scaffolds gelistet und andererseits die gefärbten Elemente innerhalb des Diagramms. Bei einem Klick auf die gelistete Problemstelle werden vom detektierten Problem betroffenen Stellen selektiert, um den selektierten Fehler im Diagramm zu erkennen.

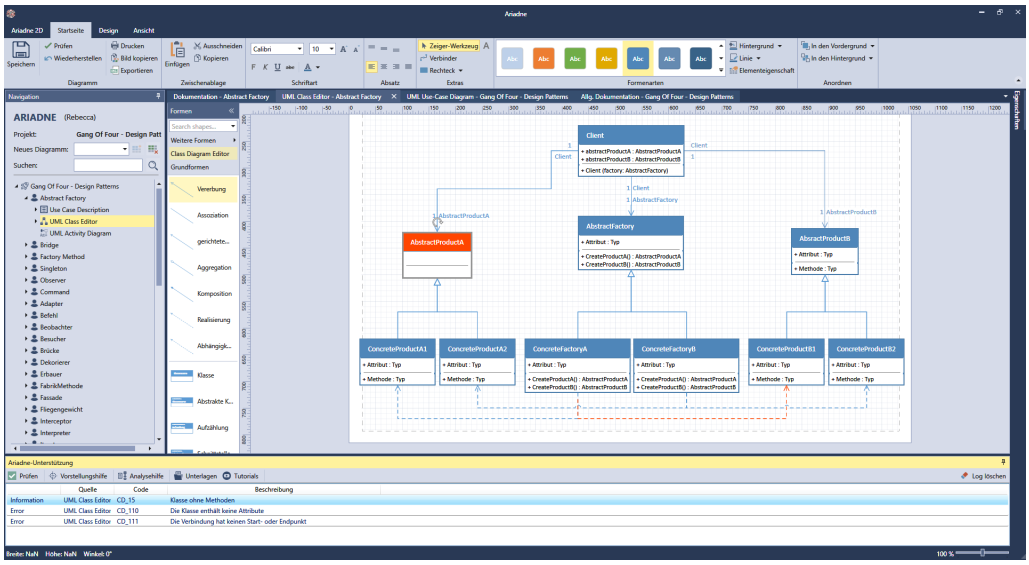


Abbildung 7.29: Markierung von Problemstellen eines Diagramms

7.3.2 *Verknüpfung Problem zu Scaffold*

Wie oben bereits erläutert, wurde für die Möglichkeit problemspezifisch Scaffolds zu empfehlen die Datenbank *AriadneRecommendation* (siehe Abschnitt 7.1.2, Abb. A.2) eingeführt. Diese verwaltet Tabellen zu identifizierten Problemen, festgelegten Regeln und Regelverstößen und ebenfalls die konkreten Inhalte der Scaffolds Unterlagen und Tutorials. Mit Hilfe der Tabelle *AriadneRecommendation* wird ein spezifisches Problem über den *Errorcode* oder *RecommendationTarget* mit der empfohlenen Hilfestellung verknüpft. Diese Konfiguration können Dozierende über den Dialog *Unterlagen* vornehmen (Abb. 7.20).

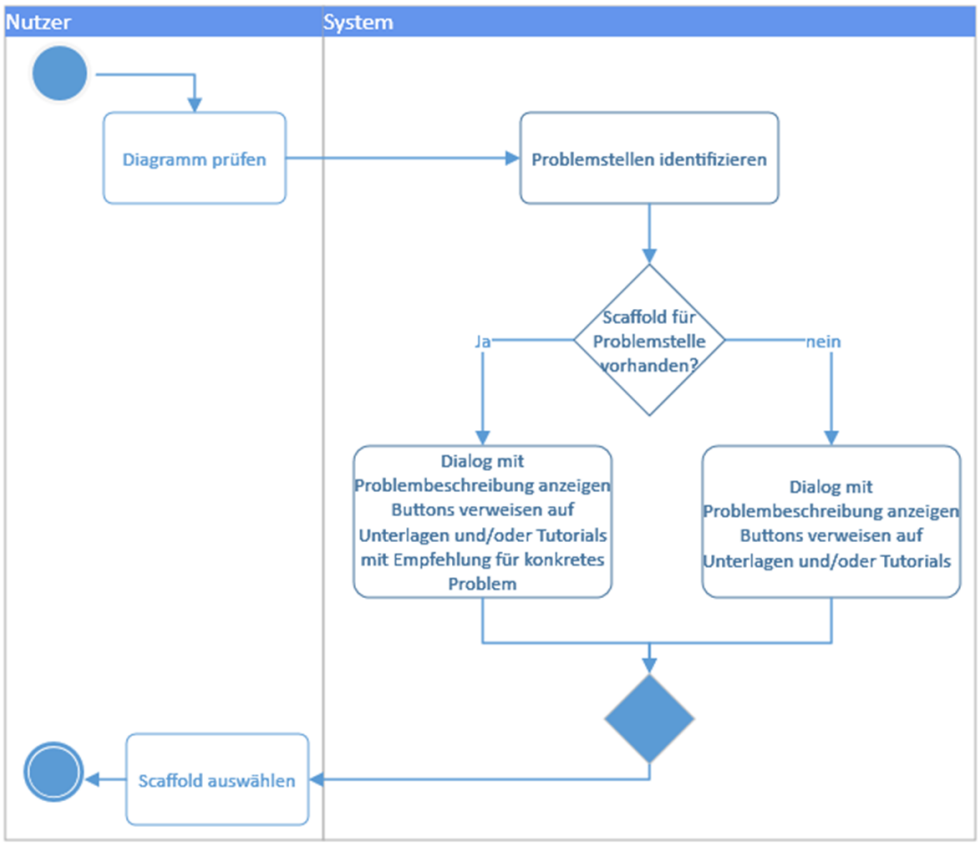


Abbildung 7.30: Aktivitätsdiagramm zur Visualisierung des Zusammenhangs zwischen einem detektierten Problem und einem empfohlenen Scaffold zu diesem

Abbildung 7.30 zeigt dazu schematisch den Ablauf der Nutzerinteraktion *Diagramm Prüfen* bis zur problemspezifischen Empfehlung. Abbildung 7.22 zeigt entsprechend zu einem Problem das empfohlene Hilfsmittel. Abbildung 7.25 zeigt einen Ausschnitt der Tabelle, die alle Problemstellen verwaltet.

7.3.3 *Bewertung von Unterlagen & Tutorials*

Die Bewertung von Unterlagen & Tutorials sieht ein drei-stufiges Konzept vor:

- Individuelle Bewertung
- Studierendenbewertung als 5-Sterne-Rating
- Dozententipp

Nutzer können eine individuelle Bewertung der einzelnen Inhalte vornehmen, beispielsweise für eine weitere/nochmalige Nutzung als Merkhilfe. Die vergebenen Ratings aller Nutzer werden zusätzlich in einem 5-Sterne-Rating erfasst, das für alle Nutzer sichtbar ist. Dieses ist mit einer priorisierten Anzeige verknüpft, sodass höher bewertete Unterlagen und Tutorials weiter oben angezeigt werden (siehe Abb. 7.24).

Dozierende haben, neben der Möglichkeit Unterlagen und Tutorials für alle Nutzer zur Verfügung zu stellen, auch die Möglichkeit einen sogenannten Dozententipp zu vergeben. Dieser wird mit Hilfe eines „Daumen-Symbols“ zusätzlich zum 5-Sterne-Rating angezeigt. Empfehlungen mit diesem Tag werden immer zuerst in der Liste der Empfehlungen angezeigt.

7.3.4 *Analysehilfen (Textanalysen)*

Die Scaffolds zur Identifikation von Use-Cases und zur Identifikation von Klassenkandidaten basieren auf der Nutzung des OpenNLP (Abschnitt 7.1.4.6, Version 1.0.6286.19007). Wie bereits in der Einführung zu NLP bzw. OpenNLP in Abschnitt 7.1.4.6 beschrieben, steht mit dem OpenNLP eine quelloffene Implementierung zur Verfügung, die für die Ariadne-Implementierung eingesetzt wird. Ebenso stehen vorgefertigte Modelle zu verschiedenen Einsatzgebieten zur Verfügung, die verwendet wurden. Für Ariadne bzw. die Analysehilfen, wurden folgende Module des OpenNLP verwendet:

- EnglishMaximumEntropyTokenizer
- EnglishMaximumEntropyTagger
- EnglishMaximumEntropySentenceDetector

Die verwendeten Modelle sind:

- EnglishSD.nbin
- GermanPOS.nbin
- GermanTOK.nbin

QC 10 zeigt beispielhaft, wie ein Paragraph in einzelne Sätze zerlegt wird.

```
private string[] SplitSentences(string paragraph){
        if (_sentenceDetector == null){
            var modelPath = @"Resources\";
            _sentenceDetector = new
             ↪  EnglishMaximumEntropySentenceDetector(modelPath +
             ↪  "EnglishSD.nbin");
        }
        return _sentenceDetector.SentenceDetect(paragraph);
    }
```

QC. 10: Zerlegen von Text in Sätze

In QC 11 ist die zentrale Methode der Substantiv-Verb-Analyse beschrieben, in der Substantive rot und Verben blau gefärbt werden können. Es werden Sätze zunächst mit Hilfe der Methode *EnglishMaximumEntropyTokenizer()* in Tokens zerlegt. Anschließend werden die einzelnen Tokens mit der Methode *PosTagTokens()* und *EnglishMaximumEntropyPosTagger()* auf Satzbestandteile hin analysiert.

Die Tokeneigenschaft „NN“ steht dabei für die Identifikation eines Substantivs, „V%“ steht für die Identifikation von Verben[48].

48 https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

```
void MarkKeywords(int index, string description){
...
        var sentence = description;
        //Tokenizing
        var tokenizer = new EnglishMaximumEntropyTokenizer("Resou
        ↪  rces/GermanTOK.nbin");
        var tokens = tokenizer.Tokenize(sentence);
        //Pos-Tagging
        var tags = PosTagTokens(tokens);
        for (int tagIndex = 0; tagIndex < tags.Length;
        ↪  tagIndex++){
            // see: https://www.ling.upenn.edu/courses/Fall_2003/
            ↪  ling001/penn_treebank_pos.html
            // detect nouns
            if (tags[tagIndex] == ("NN")){
                MarkText(((RichTextBox)GridDescription.Children[i
                ↪  ndex - 1]).Document.ContentStart,
                ↪  tokens[tagIndex], Brushes.Red);
            }
            // detect verbs
            else if (tags[tagIndex].StartsWith("V")){
                MarkText(((RichTextBox)GridDescription.Children[i
                ↪  ndex - 1]).Document.ContentStart,
                ↪  tokens[tagIndex], Brushes.Blue);
            }
        }
}
```

QC. 11: Färben der Substantive und Verben

Die Methode EnglishMaximumEntropyPosTagger, die in QC 11 aufgerufen wird, kann ein Token als Substantiv oder Verb markieren. Es können weitere Satzbestandteile oder auch Verbkonjugationen erkannt werden[48] die aber für die Substantiv-Verb-Analyse keine Rolle spielen. Zunächst wird der Tokenizer aufgerufen, den Satz in Bestandteile zu zerlegen (siehe Kommentar //Tokenizing in QC 11), anschließend werden die identifizierten Tokens mit Hilfe von POS-Tagging nach Substantiv und Verb getagged (siehe Kommentar //POS-Tagging in QC 11). QC 12 zeigt die Implementierung des POS-Taggings, die in QC 11 aufgerufen wird.

```
private string[] PosTagTokens(string[] tokens){
       var modelPath = "Resources\";
       var posTagger = new OpenNLP.Tools.PosTagger.EnglishMaximumE
        ↪  ntropyPosTagger(modelPath +
        ↪  "\GermanPOS.nbin");
       return posTagger.Tag(tokens);
   }
```

QC. 12: POS-Tagging der erkannten Tokens

Abbildung 7.31 zeigt den Dialog, die auch Nutzern respektive Studierenden bei der Nutzung dieses Scaffolds präsentiert wird.

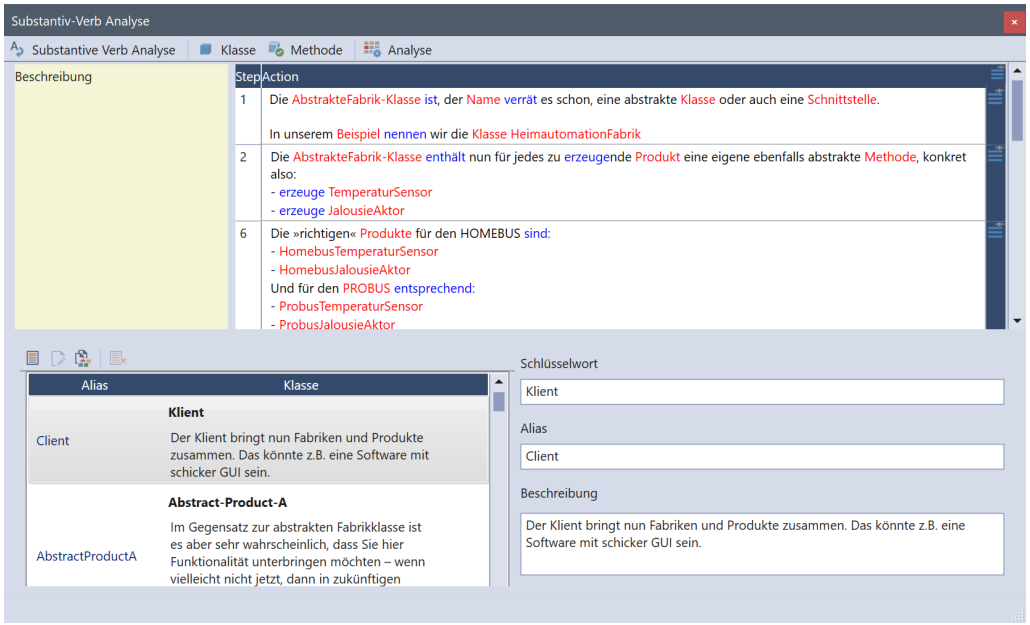


Abbildung 7.31: Beispiel einer Substantiv-Verb-Analyse

7.3.5 *Logging*

Mit Hilfe der Integration der Enterprise Library bzw. Logging Application Block der Microsoft Enterprise Library (Dominic et al., 2013) ist es möglich, verschiedene telemetrische Daten, die innerhalb von Ariadne anfallen, zu loggen. Für Ariadne wurden folgende Kategorien definiert:

- Error (interne Fehler): Diese Daten protokollieren Fehler in Ariadne.
- Warning (interne Fehler): Diese Daten protokollieren, dass ein Problem aufgetreten ist, die Ariadne beeinträchtigen oder zu einem ernsthaften Problem führen können, wenn keine Maßnahmen ergriffen werden.

- Protocol: Diese Daten protokollieren den Prozess.
- Recommendation: Diese Daten repräsentieren die Auswahl, Kontext und die Dauer der Nutzung der Scaffolds.
- Eyetracking: Diese Daten repräsentieren beispielsweise Zeitstempel und Koordinaten der Diagrammelemente.

Abhängig von den gewählten Kategorien, fallen sehr unterschiedliche Datenmengen an (für das Beispiel Eyetracking sind dies üblicherweise sehr große Datenmengen). Deshalb wurde die Wahl der Kategorien konfigurierbar realisiert und kann, abhängig von der Zielsetzung der Untersuchungen, aktiviert werden (QC 13). Alle Daten werden, sofern konfiguriert, in der Telemetrie-Datenbank abgelegt.

Mit Hilfe dieser Daten können einerseits weitere Forschungsarbeiten realisiert werden, andererseits erhalten Dozierende damit die Möglichkeit, quantitative Evaluationen zum Angebot ihrer Scaffolds durchzuführen. Es könnten beispielsweise die Verweildauer in einem Scaffold, der Zeitpunkt der Nutzung eines Scaffolds und der Stand des Diagramms evaluiert werden und so Rückschlüsse auf bestehende Probleme oder eben die Eignung eines Scaffolds gezogen werden.

```
<categorySources>
...
   <add switchValue="All" name="Error">
     <listeners>
       <add name="Database Trace Listener" />
     </listeners>
   </add>
   <add switchValue="All" name="Recommendation">
     <listeners>
       <add name="Database Trace Listener" />
     </listeners>
   </add>
   <add switchValue="All" name="Protocol">
     <listeners>
       <!--<add name="Database Trace Listener" />-->
     </listeners>
   </add>
   <add switchValue="All" name="Eye Tracker">
     <listeners>
       <!--<add name="Database Trace Listener" />-->
     </listeners>
   </add>
   <add switchValue="All" name="Warning">
     <listeners>
       <add name="Database Trace Listener" />
...
```

QC. 13: Auszug aus App.config für die Ariadne-interne Protokollierung

Folgende Auflistung erläutert wichtige Bestandteile der folgenden QC 14 bis 18:

- „LiteraturePDF - Created“ beschreibt das Ereignis.
- „ExceptionCategory.Recommendation“ betrifft die Logging-Kategorie Recommendation.
- „ExceptionPriority.Normal“ ist ein Wert über die Wichtigkeit im Betriebssystem.
- „ExceptionID.ExceptionStartID“ bestimmt die Ereignis-ID im Eventlog des Systems, auf dem die Anwendung läuft.
- „TraceEventType.Information“ bestimmt die Ebene, auf der das Ereignis eingetreten ist.

- „LiteraturePDF“ Name der Quelle, in der das Ereignis aufgetreten ist.

QC 14 bis 18 beschreiben eine Möglichkeit von der Erfassung dedizierter Logging-Parameter einer Empfehlung, bis hin zur Auswertung der Nutzungsdaten.

```
/// <summary>
/// default constructor
/// </summary>
   public LiteraturePDF()
   {
...
   Logger.Write(
    "LiteraturePDF - Created " +
     ↪ AriadneTelemetry.SerializeJson("LiteraturePDF - " +
     ↪ PDFName),
    ExceptionCategory.Recommendation, ExceptionPriority.Normal,
     ↪ ExceptionID.ExceptionStartID,
    TraceEventType.Information, "LiteraturePDF");
...
```

QC. 14: Protokollierung: Öffnen eines PDFs in Unterlagen

```
/// <summary>
/// close LiteraturePDF control
/// </summary>
    public void Close()
    {
      Logger.Write(
       "LiteraturePDF - Closed " +
        ↪ AriadneTelemetry.SerializeJson("LiteraturePDF - " +
        ↪ PDFName),
         ExceptionCategory.Recommendation,
          ↪ ExceptionPriority.Normal,
          ↪ ExceptionID.ExceptionStartID,
         TraceEventType.Information, "LiteraturePDF");
     ...
```

QC. 15: Protokollierung: Beenden eines PDFs in Unterlagen

QC 14 bis 15 beschreiben exemplarisch das Logging der Nutzungsinformationen, wenn ein PDF im Scaffold Unterlagen geöffnet bzw. geschlossen wird.

```
SELECT COUNT(*)
FROM [AriadneTelemetry].[dbo].[Recommendation]
WHERE Title Like 'LiteraturePDF' AND [Message] LIKE '%UML%'
```

QC. 16: Abfrage der genutzten Unterlagen zu *UML* im Namen der Unterlagen

QC 16 zeigt eine mögliche Abfrage in der Telemetry-Datenbank. Es wird ausgewertet, wie viele PDFs der Unterlagen mit „UML“ im Titel geöffnet und wieder geschlossen, d. h. genutzt, wurden.

```
/****** Skript für SelectTopNRows-Befehl aus SSMS ******/
SELECT COUNT(*)
FROM [AriadneTelemetry].[dbo].[Recommendation]
WHERE Title Like 'LiteraturePDF' AND [Message] LIKE '%UML%' AND
↪  [Message] LIKE '%Created%'
```

QC. 17: Abfrage der genutzten Unterlagen zu *UML* im Namen der Unterlagen und entsprechende Häufigkeit

Die Fragestellung kann weiter eingeschränkt werden, wenn z. B. „Created“ in die Datenbankabfrage mit eingebracht wird. QC 17 gibt die Anzahl der PDFs, die geöffnet wurden, aus.

```
/****** Skript für SelectTopNRows-Befehl aus SSMS ******/
(SELECT TOP 1 Timestamp
FROM [AriadneTelemetry].[dbo].[Recommendation]
WHERE Title Like 'LiteraturePDF' AND [Message] LIKE 'VL-OOA1' AND
↪  [Message] LIKE '%Closed%')
                -
(SELECT TOP 1 Timestamp
FROM [AriadneTelemetry].[dbo].[Recommendation]
WHERE Title Like 'LiteraturePDF' AND [Message] LIKE 'VL-OOA1' AND
↪  [Message] LIKE '%Created%')
```

QC. 18: Abfrage zur Nutzungszeit des PDFs *VL-OOA1*

Die Differenz der Timestamps zwischen „Created“ und „Closed“ einer Unterlage (beispielhaft in QC 18) gibt Auskunft darüber, wie lange die

erste gefundene Unterlage „VL-00A1“ genutzt wurde. Beliebige weitere Abfragen der View *Recommendation* sind denkbar.

```
private void DiagramControl_ItemsMoving(object sender,
 ↪  DiagramItemsMovingEventArgs e)
  {
    int index = 0;  // index to identify any moving item
    // for each item that is moving
    foreach (var item in e.Items) {
       var diagramItem = item.Item as DiagramItem;
       // get the index of item which is moving
       index = diagramControl.Items.IndexOf(diagramItem);
       DiagramDesignerControlEx.AriadneLogger(
        "Sequence Diagram: ItemsMoving - " +
         ↪  AriadneTelemetry.SerializeJson(
       item.Item?.CustomStyleId?.ToString() + "_" + index,
        ↪  item.NewDiagramPosition, item.OldDiagramPosition),
       ExceptionCategory.EyeTracker, ExceptionPriority.Normal,
        ↪  ExceptionID.ExceptionStartID,
        ↪  TraceEventType.Information, "Sequence Diagram");
 }
...
```

QC. 19: Logging der relevanten Eyetracking Daten für *ItemsMoving*() im Sequenzdiagramm

Als weiteres Beispiel logg QC 19 loggt jedes Ereignis, in dem ein Element in einem Sequenzdiagramm bewegt wurde und schreibt Log-Einträge in die Datenbank *AriadneTelemetry*, sofern „Eye-Tracker“ in QC 13 konfiguriert ist.

```
{
    "Name": "activation_4",
    "ExecutionTime": "07/20/2020 16:56:44",
    "Duration": null,
    "TaskName": "Ariadne",
    "ProcessId": "252",
    "DiagramElementPositionOld": "455,315",
    "DiagramElementPositionNew": "455,350",
    "DiagramElementSizeOld": "0,0",
    "DiagramElementSizeNew": "0,0",
    "DiagramZoomOld": 0.0,
    "DiagramZoomNew": 0.0,
    "DiagramPosition": "0,0",
    "DiagramElementSize": "0,0"
}
```

QC. 20: Beispiel-JSON von *ItemsMoving*()

QC 20 zeigt den JSON-Eintrag, der mit Hilfe von QC 19[49] generiert wird, mit Hilfe dessen die Daten in der *AriadneTelemetry*-Datenbank verwaltet werden.

Zusätzlich zum Logging von Bildschirmdaten und Informationen zur Nutzung von Scaffolds, wird auch ein Fehlerprotokoll für Ariadne erstellt, das mögliche Abstürze, in QC 21 bei einer Validierung eines Diagramms, protokolliert.

```
try{
    Validate();
    }
    catch (Exception exception){
        DiagramDesignerControlEx.AriadneLogger(
        "Sequence Diagram: Validating - " + exception.Message,
        ExceptionCategory.Error, ExceptionPriority.Normal,
        ↪  ExceptionID.ExceptionStartID,
        raceEventType.Error, "Sequence Diagram");
    }
```

QC. 21: Logging von Fehlern

QC 22 zeigt ein Beispiel, wie die Aktualisierung eines Dokuments in den Unterlagen protokolliert wird.

49 Dieses Beispiel zeigt, dass es prinzipiell möglich ist, Bildschirminteraktionen, die innerhalb von Ariadne getätigt werden, zu loggen.

```
public void UpdateDOC(){
    Logger.Write(
       "LiteratureDOC - UpdateDOC" + DOCName,
         ExceptionCategory.Protocol, ExceptionPriority.Normal,
          ↪  ExceptionID.ExceptionStartID,
         TraceEventType.Information, "LiteratureDOC");
...
```

QC. 22: Logging von Protokoll-Daten

Mit Hilfe der Protokollierung ausreichender Nutzungsdaten können bei einer Verknüpfung mit der Bewertung von Scaffolds durch Dozierende und Studierende, Rückschlüsse über die Relevanz und Qualität eines Scaffolds gezogen werden.

Abbildung 7.32 zeigt beispielhaft die Anzeige für Dozierende zur Auswertung der Protokolldaten. In den Reitern stehen die oben aufgezählten Kategorien zur Protokollierung (Eye Tracker, Fehler, etc.) zur Verfügung.

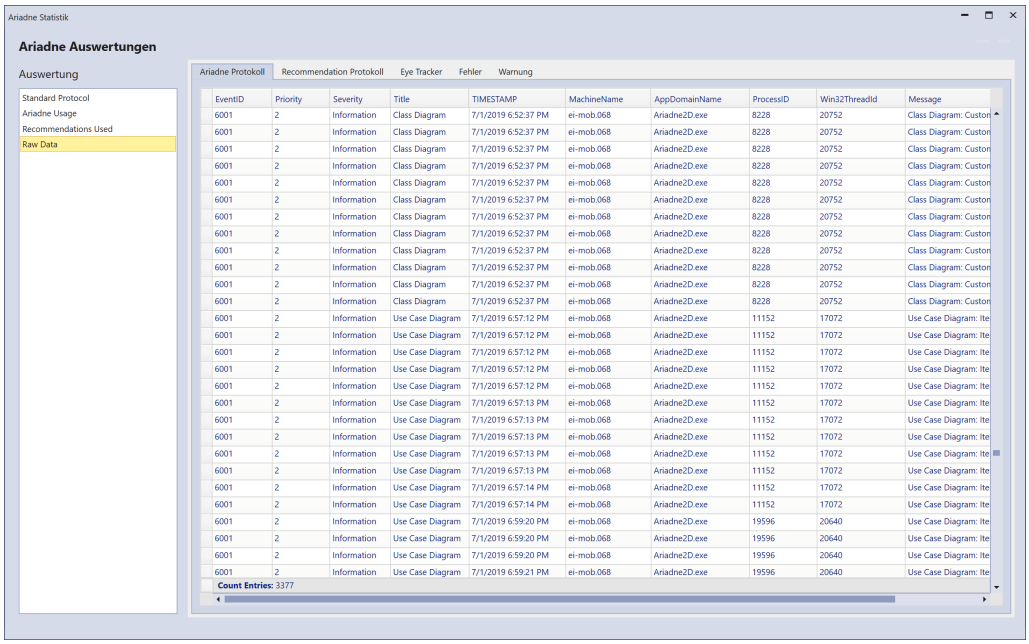


Abbildung 7.32: Beispielanzeige der geloggten Protokoll-Daten

Tabelle 7.2 zeigt relevante Felder der Datenbanksicht Recommendation aus der Telemetry-Datenbank.

Tabelle 7.2: Auszug der Felder der Datenbanksicht *Recommendation* aus der *Telemetry*-Datenbank

Datenbankfeld	Beschreibung
[Title]	Diagrammtyp, z. B. State Diagram
[TIMESTAMP]	Zeitstempel der Datenbank in UTC-Time
[ProcessID]	ID des Prozesses oder die UserID, wenn ein Nutzer der Datenerfassung im Login-Dialog (Abb. A.3) zugestimmt hat.
[ProcessName]	Name des Prozesses unter dem der Eventlog-Eintrag geschrieben wird.
[FormattedMessage]	Enthält einen JSON oder eine formatierte Nachricht mit den menschen- lesbaren formatierten Informationen

7.4 FAZIT

In diesem Kapitel wurden Einblicke in die Modellierungsumgebung Ariadne gegeben. Nach der Vorstellung der Vorüberlegungen und der Gesamtarchitektur wurde die Funktionsweise der relevanten Komponenten unter Einbezug des vorherigen Kapitels (Kapitel 6) vorgestellt: Neben der Oberfläche zur Modellierung, den Editoren und der Navigationsstruktur, wurde die Funktionsweise der klassischen Scaffolds (Abschnitt 6.1.1) näher erläutert. Außerdem wurde eine Möglichkeit vorgestellt Nutzerdaten zu loggen, um beispielsweise Daten für die Messung der Qualität von Hilfsmitteln zu sammeln.

8

EVALUATION: NUTZERSTUDIEN ZUR MODELLIERUNGSUMGEBUNG ARIADNE

Can you check the result?
Can you check the argument?
Can you derive the result differently?
Can you see it at a glance?

— George Pòlya (Polya, 1985)

Kapitel 8 betrachtet Nutzerstudien zu Ariadne, sowohl in der methodischen Vorgehensweise und dem forschungstheoretischen Rahmen des DSR wie auch in der inhaltlichen Durchführung.

Für die Arbeit ist Design Science Research (DSR) als Forschungsprozess gewählt worden, wie in Kapitel 2 vorgestellt. DSR findet breiten Einsatz im informationswissenschaftlichen Kontext, in Ingenieurwissenschaften und der Informatik. Auch im Rahmen der Didaktik ist dieser Ansatz zu finden und wird dort als Design Based Research (DBR) bezeichnet. Der Terminus Design Based Research findet sich auch unter Design Research. Beide Ansätze (DSR und DBR) haben ihre Wurzeln im Ansatz von Simon (1996). Dieser besagt, dass die Designwissenschaft ihr Potenzial erreicht, wenn innovative Artefakte geschaffen werden, die Probleme der realen Welt lösen. Beide Ansätze befassen sich mit den Forschungsprozessen, die mit dem Design und der Entwicklung von Produkten und Umgebungen verbunden sind, insbesondere in komplexen Bereichen. De Villiers und Harpur (2013) bezeichnen DBR als „the educational technology variant of design science research (DSR)“ (de Villiers & Harpur, 2013, S. 252)[50]. Mc Kenney und Reeves (2010) verallgemeinern den Begriff Design Research zu Educational Design Research (EDR) und verstehen darunter

50 Weskamp (2019) liefert hier einen guten Einblick in die Historie der Entstehungen und die Verwendung der verschiedenen existierenden Termini, die im Rahmen des Design Research verwendet werden können.

eine Familie von Ansätzen, die Theorie- und Designentwicklung hinsichtlich eines bestehenden Problems in der Praxis anstreben (Mc Kenney & Reeves, 2010, S. 17; Weskamp, 2019, S. 46). Abbildung 8.1 visualisiert die Zusammenhänge der Termini.

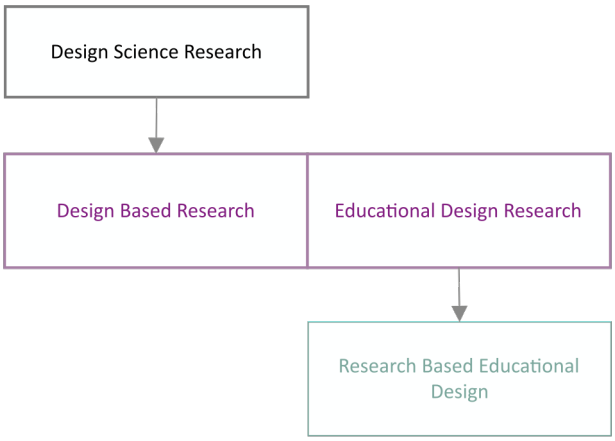


Abbildung 8.1: Einordnung der Termini zu Design-Prozessen

Ein zentrales Artefakt dieser Arbeit ist eine innovative Umgebung für die Software Engineering-Lehre. Deshalb sind auch Studierende primäre Nutzer der Software, weshalb nun als forschungstheoretische Grundlage der Evaluation eine Form des Design Based Research bzw. Educational Design Research, genauer Research-Based Educational Designs (RBED) (RBED, (Leinonen, Toikkanen & Silfvast, 2008; Mc Kenney & Reeves, 2010)), in dem ein forschungsorientiertes Design im Vordergrund steht, herangezogen wird.

Mc Kenney und Reeves (2010) führen dazu aus, dass sowohl im Educational Design Research wie auch bei RBED eher formative Evaluationen angewendet werden und summative Evaluationen in der Auswertung weniger verbreitet sind. „Im Gegensatz zu anderen Forschungsansätzen wird in Bezug auf Research-Based Design hinsichtlich der Analyse von einer Problemerkundung ausgegangen, die eher einen informellen Charakter aufweist“ (Weskamp, 2019, S. 57). In der vorliegenden Arbeit entspricht dies der Identifikation der Probleme (Abschnitt 5.3.5). „Research-Based Educational Design [...] fokussier[t] in erster Linie auf die Verbesserung der Unterrichtspraxis bzw. auf die Entwicklung eines Produkts und weniger auf die (Weiter-)Entwicklung von Theorien“ (Weskamp, 2019, S. 55; siehe auch Mc Kenney & Reeves, 2010, S. 28; Elliott, 1991, S. 49; Matter, 2017, S. 101). Eine Reflexion hat in diesem Ansatz zum Ziel, Entscheidungen bezüglich der Weiterentwicklung des Produkts zu treffen (Mc Kenney & Reeves, 2010, S. 29; Weskamp, 2019, S. 47). Diese Art von Ergebnis zeigt die praktische Relevanz der Designforschung auf.

Aus diesem Grund wird Designforschung auch als nutzungsinspirierte, anwendungsorientierte und/oder sozial verantwortliche Forschung bezeichnet (van den Akker, Bannan, Kelly, Nieveen & Plomp, 2010, S. 91; van den Akker, 1999). Designforschung ist außerdem von Natur aus iterativ angelegt, wobei jede Iteration dazu beiträgt, den Prototypen (es können auch mehrere sein), auch Inkrement (siehe Kapitel 2) genannt, zu verbessern.

Leitend für diese Arbeit war das sogenannte evolutionäre Prototyping, in dem ein Prototyp, auf Grundlage formativer Evaluationsergebnisse und der Reflexion der Entwickler des Prototyps, kontinuierlich verfeinert wird (van den Akker et al., 2010, S. 90).

In diesem Abschnitt wird das Konzept der formativen Evaluation, Einordnung und Vorgehen, wie von van den Akker et al. (2010) und Nieveen und Flomer (2013) vorgeschlagen, betrachtet und in den Kontext des Inkrements Ariadne gestellt. Was den Begriff der Evaluation betrifft, so verwendet The Joint Committee on Standards of Educational Evaluation (1994) die folgende Definition:

„Evaluation is the systematic investigation of the worth or merit of an object“ (The Joint Committee on Standards of Educational Evaluation, 1994, S. 3).

Der Verdienst (merit) bezieht sich auf den inhärenten, intrinsischen Wert des Objekts, während sein Wert (worth) als sein kontextuell bestimmter, ortsgebundener Wert definiert wird (Lincoln & Guba, 1979, S. 3; van den Akker et al., 2010, S. 92). Nach van den Akker et al. (2010) war Scriven (1972) der erste Autor, der zwischen formativer und summativer Evaluation unterscheidet aufgrund unterschiedlicher Funktionen unterscheidet. Die Funktion der formativen Evaluation ist „zu verbessern“ (to improve). Sie konzentriert sich auf das Aufdecken von Mängeln eines Inkrements während des Entwicklungsprozesses mit dem Zweck, Vorschläge zur Verbesserung zu generieren. Die Funktion der summativen Auswertung ist „zu beweisen“ (to proof). Sie wird durchgeführt, um Beweise für die Wirksamkeit der Intervention oder des Inkrements zu finden (Scriven, 1972, S. 62; van den Akker et al., 2010, S. 92). Diese wird im Rahmen dieser Arbeit nicht mehr betrachtet.

Das Inkrement Ariadne wurde als Prototyp implementiert und unterliegt den Kriterien der formativen Evaluation und der damit verbundenen iterativen Verbesserung. Formative Evaluationen dienen dem Zweck der Weiterentwicklung des Prototypen. Prototyping und Evaluationen finden auch im User Centered Design Process (siehe Abb. 6.1) in der Phase des Entwerfens und Evaluierens Anwendung. Laut DIN Deutsches Institut für Normung e. V. (2020) sollen „entworfene Gestaltungslösungen aus Benutzerperspektive evaluier[t] werden“. In Ergebnissen enthaltene

Informationen zu dieser Aktivität umfassen beispielsweise einen Prüfbericht zur Gebrauchstauglichkeit, einen Feldbericht oder einen Bericht zu Benutzerbefragungen. Ebenso wird vermittelt, dass ein Gestaltungsprozess „nicht ohne Iteration erreicht werden“ (DIN Deutsches Institut für Normung e. V., 2020, S. 15) kann. Diese Forscheraktivität wurde so verfolgt und wird in diesem Kapitel im Rahmen der Ergebnisdarstellung vorgestellt.

van den Akker et al. (2010) definieren, basierend auf dem Vergleich und der Synthese von Definitionen verschiedener Wissenschaftler (Brinkerhoff, Brethower, Nowakowski & Hluchyj, 2012; Flagg, 1990; Scriven, 1972; Tessmer, 1993), formative Evaluation im Kontext von Design Forschung als:

> „a systematically performed activity (including research design, data collection, data analysis, reporting) aiming at quality improvement of a prototypical intervention and its accompanying design principles.“ (van den Akker et al., 2010, S. 93)

Wie bereits erwähnt, benötigt ein Projekt im Rahmen der Designforschung in der Regel mehrere Iterationen, bevor eine optimale Lösung für das komplexe Problem erreicht werden kann. Jede Iteration konzentriert sich auf spezifische Forschungsfragen und benötigt ein geeignetes Forschungsdesign. Im Hinblick auf die Ausrichtung der Evaluation stellen van den Akker et al. (2010) und Nieveen und Flomer (2013) Kriterien auf, die für jede Iteration bestimmt werden (van den Akker et al., 2010, S. 94):

- Relevanz: Bedarf für die Intervention, Gestaltung basiert auf dem neuesten Stand der (wissenschaftlichen) Erkenntnisse
- Konsistenz: logische Konzeption der Intervention/ des Artefakts
- Praxistauglichkeit:
  - Erwartung: Erwartung, dass die Intervention/Artefakt in den Umgebungen, für die sie/es konzipiert und entwickelt wurde, einsetzbar ist.
  - Tatsache: Intervention/Artefakt ist in den Settings einsetzbar, für die sie/es konzipiert und entwickelt wurde.
- Effektivität:
  - Erwartung: Erwartung, dass die Anwendung der Intervention/ des Artefakts zu den gewünschten Ergebnissen führt.
  - Tatsache: Die Anwendung der Intervention/ des Artefakts führt zu den gewünschten Ergebnissen.

Mit fortschreitendem Entwicklungsgrad des Prototypen wird sich der Schwerpunkt der formativen Evaluation im Hinblick auf die oben genannten Qualitätskriterien verlagern. Während in frühen Entwicklungsstadien das Hauptaugenmerk auf der Relevanz und Konsistenz eines Prototyps liegen (Relevanz wurde bereits mit dem Problemkatalog (Abschnitt 5.3.5) aufgezeigt und soll in diesem Kapitel keine Rolle spielen), sollte in späteren Stadien die praktische Anwendbarkeit der Intervention beurteilt werden. Während der Entwicklung des Prototypen, verlagert sich der Schwerpunkt auf die tatsächliche Praxistauglichkeit und Wirksamkeit (van den Akker et al., 2010, S. 96).

Neben den oben genannten Kriterien ist nach van den Akker et al. (2010) und Nieveen und Flomer (2013) eine geeignete Wahl der Erhebungsmethode nötig. Dazu sagt van den Akker et al. (2010): „Design researchers need to select those formative evaluation methods that fit the research questions“ (van den Akker et al., 2010, S. 95).

Folgende Methoden werden vorgestellt (van den Akker et al., 2010, S. 95):

- Screening: Der Entwurf wird mit Hilfe von Checklisten zu wichtigen Merkmalen von Komponenten der prototypischen Intervention von Teammitgliedern überprüft.
- Begutachtung (Expert appraisal): Eine Gruppe von Experten verwendet den Prototypen, beispielsweise auf der Grundlage eines Leitfadens, im Rahmen einer Befragung, mit zentralen Fragen des Entwicklers (bzw. der Entwickler).
- Walkthrough: Ein Forscher und ein Vertreter der Zielgruppe gehen gemeinsam den Aufbau des Prototypen durch.
- Mikro-Evaluation: Eine kleine Gruppe von Nutzern (z. B. Lernende oder Lehrende) verwendet Teile des Prototyps außerhalb des normalen Settings (für die Nutzergruppe). Die Erhebung erfolgt durch Beobachtung und Interviews der Teilnehmer.
- Try-out: Eine begrenzte Anzahl der Nutzergruppe (z. B. Lehrende und Lernende) verwendet den Prototypen in einem für die Nutzer üblichen Setting. Wenn sich die Evaluierung auf die praktische Anwendbarkeit der Intervention bezieht, sind folgende Erhebungsmethoden üblich: Beobachtung, Befragung, Logbücher, Fragebögen; sollte die Effektivität der Intervention intendiert sein, könnten beispielsweise Tests durchgeführt werden.

Im Rahmen dieser Arbeit wurden zwei Iterationen (eine Mikroevaluation und eine Begutachtung) durchgeführt, die nach dem Vorgehen von van den Akker et al. (2010) bzw. Nieveen und Flomer (2013) eingeordnet werden und im folgenden Kapitel vorgestellt werden. Es sind weitere

Iterationen notwendig, um beispielweise das Kriterium der Effektivität nachzuweisen.

8.1 ÜBERSICHT DER DURCHGEFÜHRTEN EVALUATIONEN

Für die Konzeption und prototypische Umsetzung von Ariadne wurden einerseits Anforderungen der Umgebung im Rahmen der Erhebung der Probleme Studierender abgeleitet und andererseits die Wissensbasis konsultiert (siehe Kapitel 2). Im Rahmen formativer Evaluationen, der Evaluation von einzelnen Scaffolds, wurde Ariadne in verschiedenen Experimenten eingesetzt (siehe Abschnitt 6.10.1, Abschnitt 6.10.2 Abschnitt 6.11). Dort wurde der Einsatz einzelner Scaffolds evaluiert.

Interessant erscheinen in weiteren Iterationen der Evaluation eine Beurteilung durch möglichst konträre Nutzergruppen, Studierende und Modellierungsexperten, die das Gesamtkonzept Ariadne und dessen Möglichkeiten der Integration in die Lehre bewerten. Die Umgebung soll primär Studierenden einen erleichterten Einstieg in das komplexe Feld der Modellierung bieten, aber trotzdem professionell genug im Funktionsumfang ausgestattet sein, um ausreichende Kompetenzen zu vermitteln. Die in den beiden Iterationen identifizierten Aspekte bezüglich der Benutzbarkeit können so in eine weitere Evaluation von Ariadne einfließen und zu einer Verbesserung des Artefakts führen.

Folgende leitende Fragen ergeben sich für die Iterationen, die in diesem Kapitel vorgestellt werden. Betrachtet werden Kriterien der Konsistenz (bei der Evaluation mit Studierenden und Experten) und der Praxistauglichkeit, wobei bei der Evaluation mit Experten die *Erwartung* (van den Akker et al. (2010); „expected“) evaluiert wird, bei der Evaluation mit Studiereden die *Tatsache* (van den Akker et al. (2010); „actual“) evaluiert wird:

Iteration 1:

- Wie schätzen Studierende die Benutzbarkeit von Ariadne ein?
- Wie werden die Selbsterklärungsfähigkeit der Bestandteile, explizit der Scaffolds und deren Verwendung von Studierenden eingeschätzt?

Iteration 2:

- Wie schätzen Modellierungsexperten Benutzbarkeit und Einsetzbarkeit von Ariadne in Lehrveranstaltungen ein?
- Wie wird der Funktionsumfang von Ariadne von Modellierungsexperten beurteilt?

Für die Fragen geeignet erscheint die Methode der Mikro-Evaluation der Iteration 1 mit Studierenden und die Begutachtung durch Experten für Iteration 2. Laut van den Akker et al. (2010) und Nieveen und Flomer (2013) ist diese Art der Erhebung für „partly detailed product[s]“ (Nieveen & Flomer, 2013, S. 162; van den Akker et al., 2010, S. 96) und das vollständige Produkt geeignet.

8.2 Iteration 1: Mikro-Evaluation mit Studierenden

Für die erste Iteration der Evaluation des Prototypen, in dem es nicht Ziel war, dediziert einen Scaffold zu evaluieren, wurde die primäre Nutzergruppe, Studierende, die keine oder kaum Erfahrung in der Modellierung von Softwaresystemen haben, als Zielgruppe gewählt.

8.2.1 *Ziele*

Ziel dieses Experiments war es herauszufinden, ob die Editoren, die Ariadne zur Verfügung stellt, für Studierende nutzbar sind, d. h.,

- dass alle gesuchten Elemente intuitiv auffindbar sind.
- dass alle gesuchten Elemente intuitiv einsetzbar sind.
- dass das [Diagramm][51] angelegt und gelöscht werden kann.
- dass das [Diagramm] gespeichert werden kann.
- dass Elementzusammenhänge funktionieren
  (Elemente innerhalb eines Diagramms sollten intuitiv verknüpft werden können).

Damit soll zunächst sichergestellt werden, dass Probleme mit der Modellierung nicht auf Probleme mit der Modellierungsumgebung zurückzuführen sind. Insgesamt sollen die oben aufgeworfenen Fragen zu Iteration 1 beantwortet werden.

8.2.2 *Design*

Das Design der Studie ist als Mikro-Evaluation angelegt und wird mit Hilfe der bereits vorgestellten Think-Aloud-Methode (Abschnitt 5.3.4.1) durchgeführt. Mikro-Evaluationen werden außerhalb des „natürlichen“ (bekannten) Übungssettings durchgeführt. Aufgrund des einmaligen Messzeitpunkts handelt es sich um eine Querschnittstudie.

51 *Diagramm* stellt eine Variable dar, die mit dem je zu modellierenden Diagrammtyp ersetzt wird.

8.2.3 *Stichprobe*

Als Stichprobe sind ca. *N=5..10* Studierende aus dem Sommersemester 2019 geplant, die das Modul Software Engineering belegen. Die Studierenden erhalten vorab allgemeine Informationen zum Experiment und haben keine bzw. wenig Modelliererfahrung. Es erfolgt eine randomisierte Zuordnung. Die veranschlagte Stichprobe ergibt sich aus der Aussage von Nielsen und Landauer (1993), die empirisch belegten, dass bereits fünf Experten in der Lage sind 85% der Usability-Probleme in einer Software zu finden. Natürlich sind Studierende hier nicht als Experten zu bewerten, jedoch sind sie die primären Nutzer des Werkzeugs und damit die wichtigste Nutzergruppe.

8.2.4 *Instrumente*

Folgende Instrumente kommen zum Einsatz:

- Ariadne zum Modellieren der Diagramme
- 2 Aufgabenstellungen zur Modellierung:
  - Warm-Up (vorgegebenes Diagramm)
  - Szenario-basierte Aufgabe: Das vorgegebene Diagramm wurde bewusst mit Hilfe eines anderen Editors modelliert, sodass Nutzer nicht durch das Layout beeinflusst werden.
- Tutorial zu Think-Aloud:
  - kurzes Video[52] und Hinweise zum Vorgehen (Anhang A.4)
- Leitfragen:
  - Fehlen wesentliche Elemente in der Toolbox?
  - Ist das Tool Ihrer Meinung nach einfach handhabbar?
  - Ist der Diagrammbaum sinnvoll und verständlich aufgebaut?
  - Was erwarten Sie hinter dem Button Literatur?
  - Was erwarten Sie hinter dem Button Vorstellungshilfe?
  - Was erwarten Sie hinter dem Button Analysehilfe?
  - Was erwarten Sie hinter dem Button Anleitungen?
  - Allgemeines Feedback/Verbesserungsvorschläge
- Audioaufnahme
- Screencast des Bildschirms

52 https://www.youtube.com/watch?v=BwpPIiBKocA

8.2.5 *Durchführung*

1. Aufklärung über Aufnahme und Einverständnis. Fragen zu vorhandener Modelliererfahrung und bekannter Tools.
2. Tutorial zu Think-Aloud und Hinweise zum Einsatz während des Experiments.
3. Jeder Studierende erhält eine Warm-up Aufgabe und modelliert ein auf Papier vorgegebenes *Diagramm* mit dem *Editor*[53],[54].
4. Jeder Studierende erhält ein Szenario mit der Aufgabe ein (zu Schritt drei verschiedenes) *Diagramm* zu modellieren.
5. Die Teilnehmer werden gebeten, dem Experimentleitenden[55] alle Schritte, die sie vorhaben bzw. gerade durchführen, laut auszusprechen (Think-Aloud).
6. Nach der Bearbeitung der Aufgaben bespricht der Experimentleitende[55] mit dem Studierenden die wichtigsten Auffälligkeiten während der Toolnutzung unter Nutzung der Leitfragen.

Während der Modelliertätigkeiten (ab Schritt 3) werden Audioaufnahmen sowie ein Screencast der Modellierungsumgebung erstellt.

8.2.6 *Auswertung & Interpretation der Ergebnisse*

Die Auswertung dieser Iteration dient der Verbesserung des Prototypen, mit dem Fokus auf der Baumstruktur als primäre Navigation und der Editoren zur Modellierung. Audio- und Videomaterial wurden mit Hilfe von MAXQDA (2018)[56] codiert. Eine Transkription war nicht notwendig. Das Material wird aus den oben genannten Fragen abgeleitet und im Hinblick auf Benutzbarkeit und Selbsterklärungsfähigkeit der Scaffolds ausgewertet. Als Auswertungsmethode erscheint die bereits beschriebene strukturierende qualitative Inhaltsanalyse nach Mayring (Abschnitt 5.3.4.3) geeignet, um „bestimmte Aspekte aus dem Material herauszufiltern, unter vorher festgelegten Ordnungskriterien einen Querschnitt durch das Material zu legen oder das Material aufgrund bestimmter Kriterien einzuschätzen“ (Mayring, 1991, S.115). Zentral war dabei die Frage, ob Studierende Ariadne als benutzbar einschätzen und vor allem wo noch Verbesserungsbedarf im Hinblick darauf besteht. *N=12* Studierende haben an dieser Mikro-Evaluation teilgenommen. Da die

53 *Editor* stellt eine Variable dar, die mit dem je zu verwendenden Editor ersetzt wird.
54 Es wird kein Tutorial vorab zur Verfügung gestellt, um die Intuitivität und Nutzbarkeit bei Erstverwendung des Prototypen zu testen.
55 Der Experimentleitender ist in diesem Fall der Autor dieser Arbeit.
56 MAXQDA ist eine Software zur Auswertung qualitativer Daten. https://www.maxqda.de/

Evaluation zur Verbesserung des Prototypen dient und als formative Evaluation klassifiziert wurde, scheint diese Größe der Stichprobe ausreichend.

Der Begriff der Benutzbarkeit geht einher mit der Usability, Gebrauchstauglichkeit, eines Produkts, hier des Prototypen. Dieser Begriff ist in der Norm DIN EN ISO 9241 bezeichnet als „das Ausmaß, in dem ein Produkt, ein System oder Dienst durch bestimmte Nutzer in einem bestimmten Nutzungskontext genutzt werden kann um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen“ (DIN Deutsches Institut für Normung e. V., 2006). In Abschnitt 9241-11 werden sieben Grundsätze der Dialoggestaltung aufgeführt, die zur Auswertung als initiales Codesystem herangezogen wurden. In der Beschreibung der Grundsätze wird dann zwar von Webseiten gesprochen, es wird aber angenommen, dass diese prinzipiell auch für Dialoge der Software herangezogen werden können. Das Material wurde anhand dieser Kriterien codiert, respektive bewertet und aufgrund der Bewertung notwendige Verbesserungen im Prototypen vorgenommen. Der Grundgedanke bei der Auswertung des Materials auf Basis dieser Kriterien war, dass diese Kriterien für gebrauchstaugliche Software stehen, die den Prototypen verbessern.

Die sieben Grundsätze der Dialoggestaltung lauten (DIN Deutsches Institut für Normung e. V., 2006):

- Aufgabenangemessenheit: „Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen.“
- Selbstbeschreibungsfähigkeit: „Ein Dialog ist selbstbeschreibungsfähig, wenn jeder einzelne Dialogschritt durch Rückmeldung des Dialogsystems unmittelbar verständlich ist oder dem Benutzer auf Anfrage erklärt wird.“
- Erwartungskonformität: „Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z. B. den Kenntnissen aus dem Arbeitsgebiet, der Ausbildung und der Erfahrung des Benutzers sowie den allgemein anerkannten Konventionen.“
- Fehlertoleranz: „Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand durch den Benutzer erreicht werden kann.“
- Steuerbarkeit: „Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist.“

- Individualisierbarkeit: „Ein Dialog ist individualisierbar, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe, individuelle Vorlieben des Benutzers und Benutzerfähigkeiten zulässt."
- Lernförderlichkeit: „Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet."

Mit Hilfe dieser Kriterien können ebenfalls die beiden Forschungsfragen für diese Iteration beantwortet werden: Die Benutzbarkeit wird mit Hilfe aller sieben Kriterien bewertet, die Selbsterklärungsfähigkeit ist bereits ein Kriterium innerhalb der Grundsätze (Selbstbeschreibungsfähigkeit). Diese wurde vor allem für die Kategorien der Hilfsmittel evaluiert.

Zu der nun folgenden Auswertung hinsichtlich der Einordnung in die vorgestellten Grundsätze sei angemerkt, dass explizit auch problematische Stellen (in den Tabellen) vorgestellt werden, um das angestrebte Ziel der Verbesserung zu erreichen. Einige Punkte werden mehr als einem Grundsatz zugeordnet, da mehrere Grundsätze tangiert werden können. Für alle präsentierten Tabellen gilt folgende Struktur: Im allgemeinen werden die Findings (Spalte 1) beschrieben, die zur jeweilige Kategorie identifiziert wurden. In der zweiten Spalte ist vermerkt, ob dieses Finding als Problem aufgetreten ist, als Vorschlag von einem Teilnehmer abgegeben wurde oder eine Frage dazu aufgekommen ist. Auf eine quantitative Bewertung der Findings für die Problemstellen wurde verzichtet, da alleine das Auftreten eines Problems dazu führt, dass diese Stellen betrachtet werden müssen und eine zahlenmäßige Aufarbeitung keinen weiteren Erkenntnisgewinn liefert. Vorschläge wurden alle aufgenommen. In Spalte drei ist, wenn vorhanden, die Lösung zu der Problemstelle vermerkt oder eine Anmerkung, wenn beabsichtigt keine Lösung realisiert wurde. Sollte dieses Feld nicht ausgefüllt sein, existiert (noch) keine Lösung oder Strategie. Neben der tabellarischen Darstellung erfolgt eine Erläuterung zu den zentralen Findings.

8.2.6.1 *Aufgabenangemessenheit*

„Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen" (DIN Deutsches Institut für Normung e. V., 2006). Zu dieser Kategorie wurden keine Codes vergeben. Es können keine Aussagen dazu abgeleitet werden.

8.2.6.2 *Selbstbeschreibungsfähigkeit*

DIN Deutsches Institut für Normung e. V. (2006) definieren einen Dialog als selbstbeschreibungsfähig, „wenn jeder einzelne Dialogschritt durch

Rückmeldung des Dialogsystems unmittelbar verständlich ist oder dem Benutzer auf Anfrage erklärt wird“ (DIN Deutsches Institut für Normung e. V., 2006).

Um Nutzer darin zu unterstützen sich in einer Software zurechtzufinden, werden in DIN EN ISO 9241 vier Punkte (Orientierung, Antizipierbarkeit, Feedback und Hilfe) genannt, um einen Nutzer darin zu unterstützen: Zur Orientierung sollen dem Nutzer Orientierungspunkte angeboten werden, damit dieser die Möglichkeit erhält, zu erkennen, wo z. B. in welcher Hierarchie er sich befindet. Ein Dialog ist antizipierbar, wenn ein Nutzer zielsicher navigieren kann, d. h. ein Nutzer muss Navigationselemente erkennen können und bewerten, wo diese hinführen. Ein weiteres Kriterium ist Feedback, das bedeutet, dass ein Nutzer eine Rückmeldung zur durchgeführten Aktion erhält und damit ein Gefühl der Sicherheit erzeugt wird. Ebenso zu diesem Punkte zählt die Möglichkeit, dass ein Benutzer Hilfestellungen zur Lösung der Aufgaben erhält (DIN Deutsches Institut für Normung e. V., 2006).

Tabelle 8.1 beschreibt die Findings, die zur Kategorie der Selbstbeschreibungsfähigkeit identifiziert wurden.

Tabelle 8.1: Findings aus der Mikroevaluation zum Usability Grundsatz der Selbstbeschreibungsfähigkeit

Code	Problem(x)/ Vorschlag(v)	Beschreibung der Lösungsstrategie
Eigenschaftenfenster	x	Über Dropdown Menü erreichbar
Unterscheidung der Elemente Entscheidung und Zusammenführung (Aktivitätsdiagramm)	x	default-Anzeige mit Elementbeschriftung
Teilnehmer(TN) sucht nach Multiplizität beim Pfeilende und durch Rechtsklick dann Eigenschaften	x	Über Dropdown Menü erreichbar
Typ einer Klasse ändern (z. B. Abstrakt)	v	kein Problem
TN sucht das Start Element	x	default-Anzeige mit Elementbeschriftung
TN sucht Endzustand	x	default-Anzeige mit Elementbeschriftung

Code	Problem(x)/ Vorschlag(v)	Beschreibung der Lösungsstrategie
Unterschied zwischen Entscheidung und Zusammenführung	x	default-Anzeige mit Elementbeschriftung
TN findet Feld für Beschriftung einer Transition nicht	x	Über Dropdown Menü erreichbar
TN findet Lebenslinie nicht	x	default-Anzeige mit Elementbeschriftung
TN ist sich unsicher, ob es die Lebenslinie ist	x	default-Anzeige mit Elementbeschriftung
Pfeilmodus, dann Punkte bei Elementen anzeigen und dann verbinden	v	Verbindermodus
Erstellen des neuen Artefakts bzw. Editors in den Reitern	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
unbeschriftete Nachricht automatisch synchrone Nachricht	x	Benennung ändern

Im Folgenden werden die zentralen Findings und Vorschläge der Teilnehmer interpretiert und Umsetzungen erläutert:

- Teilnehmer haben das Menü, in dem Eigenschaften einer Assoziation näher definiert werden können, nicht wahrgenommen (siehe Abb. 8.2). Dafür wurde die identische Funktion im Dropdown-Menü, das über einen Rechtsklick auf das Element erreichbar ist, hinterlegt.
- Das Antragen der Multiplizität in einem Klassendiagramm wird über den entsprechenden Eintrag in den Eigenschaften der Assoziation realisiert. Analog funktioniert dies für Transitionen im Zustandsdiagramm. Teilnehmer versuchen dieses Fenster über einen Rechtsklick auf die Assoziation zu erreichen. Diese Funktion wurde im Anschluss an die Evaluation integriert. Zum Zeitpunkt des Tests musste das Fenster der Eigenschaften explizit aufgeklappt werden.
- Teilnehmer konnten die beiden Elemente des Aktivitätsdiagramms „Entscheidung“ und „Zusammenführung“ nicht unterscheiden, da sie das gleiche Aussehen aufweisen. Deshalb wurde per default,

die Beschriftung der Elemente angezeigt. Im getesteten Prototypen musste diese ausgeklappt werden.

- Die Suche nach Elementen (z. B. Endzustand im Zustandsdiagramm, Lebenslinie im Sequenzdiagramm etc.) wurde vereinfacht, in dem die Elemente der Toolbox per default mit sichtbarer Beschriftung angezeigt werden.
- Zu dem Vorschlag mehrere Stereotypen anzubieten, wurden der Toolbox weitere Elemente hinzugefügt. Es können die Stereotypen „abstract“,„interface“ und „enum“ ausgewählt werden.
- Zum Verbinden zweier Elemente durch eine Assoziation wurde ein „Pfeilmodus“ vorgeschlagen. Aufgrund weiterer auftretender Findings in den folgenden Grundsätzen, wurde ein Verbindermodus (Abb. 8.3) realisiert, der so aktiviert, Verbindungspunkte zwischen Elementen anzeigt und so eine weitere Möglichkeit der Verbindung zwischen Elementen zulässt.
- Das Anlegen eines neuen Artefakts suchen Teilnehmer in den Reitern der Oberfläche und nicht in der Baumstruktur. Deshalb wurden weitere Einstiegspunkte zum Anlegen realisiert. Zum Einen ist es möglich, ein Diagramm oberhalb der Baumstruktur hinzuzufügen, zum anderen weiterhin in der Baumstruktur über den Rechtsklick. Außerdem können Projekte und Use-Cases nun in den Reitern hinzugefügt werden.
- Im Sequenzdiagramm war der Nachrichtentyp der „synchronen Nachricht“ nicht als solcher gekennzeichnet. Die Benennung wurde vervollständigt.

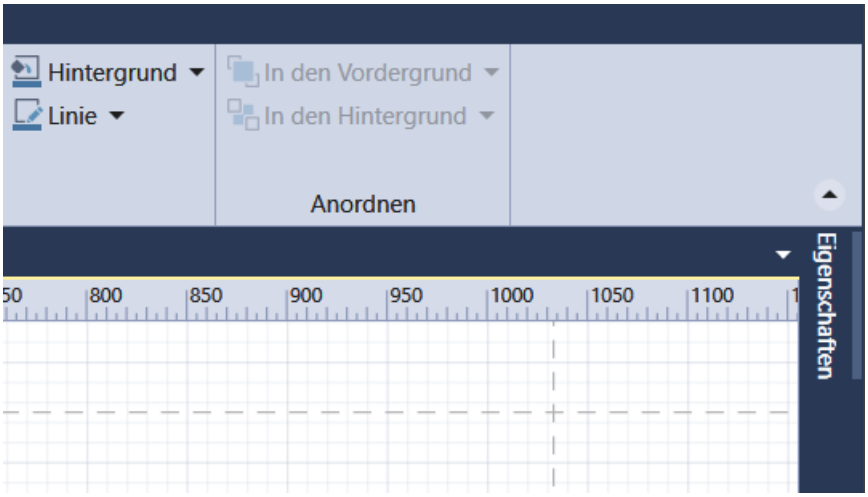


Abbildung 8.2: Platzierung des Menüs *Eigenschaften*

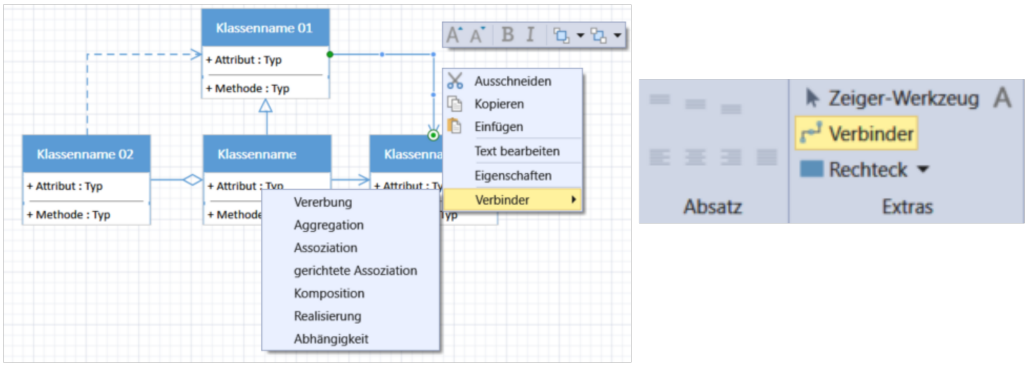


Abbildung 8.3: Verbindermodus im aktuellen Stand des Prototypen

8.2.6.3 *Erwartungskonformität*

Die DIN Deutsches Institut für Normung e. V. (2006) definieren einen Dialog als erwartungskonform, „wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z. B. den Kenntnissen aus dem Arbeitsgebiet, der Ausbildung und der Erfahrung des Benutzers sowie den allgemein anerkannten Konventionen“ (DIN Deutsches Institut für Normung e. V., 2006). Dieses Kriterium wird durch die Kriterien der Konsistenz und der Erfahrung bewertet. Ein Dialog sollte demnach möglichst viele bekannte Muster enthalten und beispielsweise Merkmale zur Navigation nicht variieren. Mit Hilfe einer konsistenten Nutzerführung sollen sich gerade Novizen nicht mit dem „Erkennen und Speichern von Mustern“ (DIN Deutsches Institut für Normung e. V., 2006) beschäftigen, sondern vielmehr ihren Fokus auf die Modellierung selbst legen können. Ebenso spielt das Kriterium der Erfahrung des Nutzers eine große Rolle. Studierende haben bereits Erfahrungen in der Verwendung verschiedener Tools gesammelt und haben deshalb Vorstellungen und Erwartungen zu bestimmten Funktionsweisen, wie Navigation, Abläufen und Benennungen von Symbolen (DIN Deutsches Institut für Normung e. V., 2006). Auch Analogien, die zu anderer Software hergestellt werden, haben hier aus Sicht des Autors Einfluss.

Gerade im Sinne der Erfahrungen, wurden Studierende dazu befragt, welche Erwartungen bzw. welche Funktionalität sie bezüglich der Buttons, die zu den verschiedenen Scaffolds führen, haben und diese diskutiert und hinterfragt. Außerdem wurde der Hinweis vermittelt, dass diese eine Form der Unterstützung liefern.

Hinter dem Button „Literatur“ erwarteten Studierende:

- schriftliche Erklärungen zum Vorgehen bei der Modellierung
- eine Suche für Befehle (z. B. Aktion) mit dem Ergebnis einer Erklärung und einem passenden Beispiel

- konkrete Unterstützung zum gerade bearbeiteten Diagrammtyp
- dass Literatur, die selbst gefunden wurde, dort abgespeichert werden kann
- Literatur, die analog zu Citavi[57] in einer Datenbank abgelegt wird
- eine Auflistung zu Büchern zum Thema UML
- Bücher zu gutem UML Stil

Diese erste Einschätzung der Studierenden deckt sich mit der Intention des Hilfsmittels, der Bereitstellung von Unterlagen. Allerdings wurde mehrfach die Benennung des Hilfsmittels „Literatur“ diskutiert. Vorschläge zur Benennung waren „Dokumentation“, „Passende oder Hilfreiche Unterlagen“, „HowTos“ und „Help“. Zudem wurde das Symbol kritisch betrachtet und Analogien zu Citavi gezogen, die vom Autor nicht intendiert waren.

Basierend auf diesen Findings wurde das Symbol dieser Hilfestellung geändert um Nutzer nicht fälschlicherweise in die Erwartung zu versetzen, dass eine Verbindung zu Citavi geschaffen wird. Zudem wurde die Hilfestellung „Literatur“ umbenannt in „Unterlagen“. Abbildung 8.4 zeigt die Symbole und Benennungen der Scaffolds im Prototypen nach den Evaluationen.

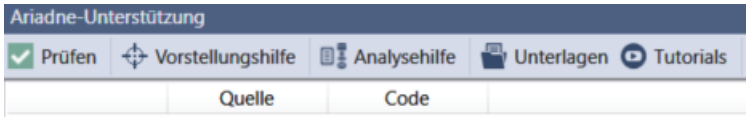


Abbildung 8.4: Integrierte Scaffolds im aktuellen Stand des Prototypen

Hinter dem Button „Anleitungen“ erwarteten Studierende:

- Video-Tutorials zur Software
- eine Schritt-für-Schritt Anleitung zu Diagrammen mit Video
- Videoanleitungen
- eine Anleitung zum Tool oder ähnliches wie hinter dem Button Literatur

Diese Einschätzung der Studierenden deckt sich mit der primären Intention des Hilfsmittels der Bereitstellung von Tutorials. Diese Hilfestellung bietet zusätzlich zu Videos einen Browser zu Online-Material an. Allerdings wurde mehrfach die Benennung des Hilfsmittels „Anleitungen“ diskutiert. Vorschlag war hier die Umbenennung zu „Tutorials“.

Hinter dem Button „Analysehilfe“ erwarteten Studierende:

- Vorschläge zur Optimierung des Diagramms.

57 https://www.citavi.com/de

- etwas, was bei der Textanalyse hilft.
- Hilfe, die Kommunikation genauer zu beschreiben.
- ein System, das Diagramme überprüft und Fehler anzeigt, wobei fehlerhafte Stellen markiert werden sollten.
- eine Funktionalität, die das aktuelle Diagramm überprüft und Hinweise auf beispielsweise fehlende Verbindungen oder unnötige Verbindungen gibt.
- ähnlich zu einem Debugger für Quellcode, die Anzeige offener und problematischer Stellen (beispielsweise, wenn Pfeile nicht verbunden sind).

Diese Einschätzung der Studierenden deckt sich nicht mit der Intention des Hilfsmittels der Bereitstellung von Hilfestellungen zur Textanalyse. Diese Erwartung hatte nur ein Teilnehmer. Die übrigen Teilnehmer erwarteten Funktionalität, die zwar in der Software realisiert ist, jedoch nicht im Rahmen dieser Hilfestellung. Zur Verbesserung des Prototypen müssen demnach noch Konzepte erarbeitet werden, um die Erwartung zu diesem Button zu verbessern.

Zum Button Vorstellungshilfe hatte keiner der Teilnehmer eine Vorstellung bzw. Erwartung. Ähnlich zur Analysehilfe müssen noch Konzepte erarbeitet werden, um die Erwartung zu diesem Button zu verbessern.

Tabelle 8.2 zeigt die Findings, die unter dem Grundsatz der Erwartungskonformität zusammengefasst wurden. Im Gegensatz zur zuvor vorgestellten Kategorie der Selbstbeschreibungsfähigkeit (siehe Tabelle 8.1), wurde zusätzlich erfasst, wenn Studierende Rückfragen zu Funktionalitäten gestellt haben. Auffällig ist zudem, dass die Anzahl der Vorschläge im Gegensatz zu den Problemstellen überwiegt.

Zu den Fragen, die Studierende während der Verwendung von Ariadne gestellt haben, zählen:

- Passt sich die Größe eines Elements dynamisch an den Textinhalt an?
- Werden Datentypen automatisch erkannt?
- Werden in der Analysehilfe Klassen vorgeschlagen, die übernommen werden können?
- Sind Kommentare im Diagramm möglich?
- Ist eine Volltextsuche in den Quellen möglich oder wird ausschließlich das offene Dokument durchsucht?
- Wie kann ein Dokument geschlossen werden?

Der Experimentleitende und Autor der Arbeit beantwortete den Studierenden die Fragen während der Durchführung:

- Die Größe eines Elements wird zum Teil dynamisch bis zu einer bestimmten Länge angepasst oder es findet ein Textumbruch statt.
- Datentypen werden nicht automatisch erkannt, was aber auch nicht intendiert war, da auch die Bestimmung geeigneter Datentypen ein Lernziel darstellen kann.
- Die vorgeschlagenen Use-Cases, Akteure und Klassenkandidaten können per Drag-and-Drop in das Diagramm übernommen werden.
- Kommentare, direkt im Diagramm, können über Textfelder hinterlassen werden. Eine Kommentarfunktion wie im Texteditor gibt es nicht.
- Eine Volltextsuche ist sowohl im Dokument wie auch über alle Dokumente möglich.

Im Folgenden werden die zentralen Findings und Vorschläge der Teilnehmer interpretiert und Umsetzungen erläutert:

- Teilnehmer bemängeln, dass Elemente, so sie per Doppelklick auf die Modellierungsfläche gebracht werden, immer zentral in der Mitte der Fläche angezeigt werden. Dies wurde bisher nicht umgesetzt.
- Es wurde vorgeschlagen, Kopien von Elementen dediziert an Stelle des aktuellen Mauszeigers platzieren zu können. Dieser Vorschlag wurde umgesetzt.
- Für die Modellierung von Assoziationen wurde ein neues Bedienkonzept vorgeschlagen: Zunächst soll der Pfeiltyp ausgewählt werden, im Anschluss die Quelle und schließlich das Ziel. Dieser Vorschlag wurde nicht umgesetzt, da eben ein vollständig neues Bedienkonzept realisiert werden müsste.
- Das Bedienkonzept Drag-and-Drop stellt nicht alle Teilnehmer zufrieden. Hier wurde ebenfalls kein neues Bedienkonzept entwickelt, da bereits das alternative Konzept des „Doppelklicks“ auf Elemente der Toolbox unterstützt wird.
- Für das Anlegen eines neuen Projekts sowie eines neuen Diagramms, wurden weitere Möglichkeiten oberhalb der Baumstruktur für Diagramme und unter dem Reiter Ariadne 2D für neue Projekte geschaffen, da die Funktion über Rechtsklick auf das Projekt bzw. den Klassendiagramm-Editor eines Projekts nicht gefunden wurde (siehe Abb. 8.5).
- Für Empfehlungen von Hilfestellungen wurde vorgeschlagen, diese direkt im Diagramm anzuzeigen. Dieser Vorschlag wurde nicht umgesetzt, da hier neue Schwierigkeiten beispielsweise in der Übersicht des Diagramms vermutet wurden und andere Teilnehmer die Präsentation unter den Hilfsmitteln positiv empfanden.

- Wegen des Wunsches bzw. Vorschlags, einen Pfeil automatisch andocken zu können, wurde ein Verbindermodus realisiert, der, wenn aktiviert, ebenso Ankerpunkte an den Elementen anzeigt.
- Der Vorschlag Pfeile anders einzufärben, die nicht korrekt an Elemente gebunden sind, wurde in der Funktionalität „Diagramm Prüfen“ der Problemerkennung (siehe Abschnitt 7.3) implementiert.
- Alle weiteren nicht explizit genannten, nicht ausgefüllten Felder der dritten Spalten sind derzeit noch nicht umgesetzt, können aber in zukünftigen Versionen realisiert werden.

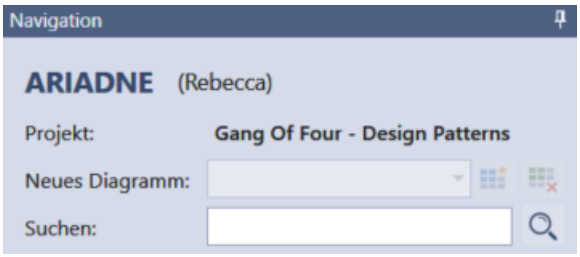


Abbildung 8.5: Möglichkeit zum Anlegen von Diagrammen

Tabelle 8.2: Findings aus der Mikroevaluation zum Usability Grundsatz der Erwartungskonformität

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Elemente werden, wenn per Doppelklick ausgewählt in der Mitte der Modellierungsfläche angezeigt	x	
Einfügen einer Kopie an Stelle des Mauszeigers	v	umgesetzt
Anwählen des gewünschten Pfeiltyps und anschließende Wahl von Quelle und Ziel	v	
Anlegen eines neuen Projekts unter Start oder Einfügen vermutet	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
neues Artefakt Hinzufügen über Reiter Einfügen	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
neues Diagramm über Rechtsklick	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
Empfehlungen direkt im Diagramm nicht unten	v	
Vorwärts /Rückwärtsschritt	v	umgesetzt: Rückwärtsschritt Strg+Z
Button auf Startseite welches Diagramm ausgewählt werden soll	v	
Bei Rechtsklick auf Element Optionen anzeigen	v	umgesetzt
Enter erzeugt Zeilenumbruch, gewünscht Bestätigung	v	
Beispieltext löschen bei Klick in das Textfeld	v	
Attribute standardmäßig mit „-“ beginnen	v	
Markieren des gesamten Texts zur Vergabe eines Namens	v	umgesetzt
Anzeige der Abstände zwischen Elementen	v	umgesetzt
Export als SVG	v	umgesetzt
Drag-and-Drop	x	keine Problemlösung, Doppelklick auf Element bietet alternative Funktion
Auswahldropdown-Menü mit Sichtbarkeiten und Rückgabetyp (Analogie: Word Referenzen)	v	

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Nachrichtenpfeile könnten beim Ziehen auf den Editor gleich mit dem Pfeil nach rechts und waagrecht angezeigt werden (Sequenzdiagramm)	v	umgesetzt
Antwortnachrichten könnten von rechts nach links waagrecht dargestellt werden (Sequenzdiagramm)	v	umgesetzt
Zeilenumbruch nach „/“ bei Transition (Zustandsdiagramm)	v	
Symbole zu klein mit Beschreibung besser	v	umgesetzt
rote Unterringelung („String“ nicht deutsch) störend	v	umgesetzt
Pfeil automatisch andocken	v	Verbindermodus
Pfeil anders einfärben, wenn er angedockt ist, wie wenn er nicht verbunden ist	v	umgesetzt während Diagramm Prüfen
Ankerpunkt bereits beim hovern mit dem Pfeil finden	v	Verbindermodus
Anzeige Lebenslinie über Aktivierungsbalken	x	Feature „in den Vordergrund, in den Hintergrund“
automatisches Alignment	v	

Insgesamt wurden von den Teilnehmern ein positives Fazit gezogen und alle würden es wiederverwenden:

- Das Design der Oberfläche in Analogie zu Office-Produkten wurde positiv gewertet.

- Überwiegend erleichternd wurde die Bedienung und die individuelle Definition von Attributen, Methoden sowie Assoziationen (Transitionen etc.) bewertet.
- Shortcuts wurden ebenso als positive Eigenschaft in der Bedienung hervorgehoben.

8.2.6.4 *Fehlertoleranz*

Die DIN Deutsches Institut für Normung e. V. (2006) beschreibt einen Dialog als fehlertolerant, „wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand durch den Benutzer erreicht werden kann“ (DIN Deutsches Institut für Normung e. V., 2006).

Für diesen Grundsatz wurden lediglich Bugfixes realisiert.

8.2.6.5 *Steuerbarkeit*

Laut DIN Deutsches Institut für Normung e. V. (2006) gilt ein Dialog als steuerbar, „wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist“ (DIN Deutsches Institut für Normung e. V., 2006).

Tabelle 8.3 beschreibt die Findings für die Kategorie Steuerbarkeit.

Tabelle 8.3: Findings aus der Mikroevaluation zum Usability Grundsatz der Steuerbarkeit

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Pfeile entlang Aktivitätsbalken verschieben (Sequenzdiagramm)	v	
Element markieren und so oft wie möglich einfügen	v	umgesetzt
Bei Klick auf Entfernen, Schere anzeigen	v	
„Text bearbeiten“ könnte auch „Text einfügen“ heißen	v	
verschiedene Ankerpunkte für die Nachrichten am Aktivierungsbalken	v	umgesetzt

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Ankerpunkte verschiebbar	v	
Shortcuts für Klassen	v	
Hilfebutton oben rechts oder extra Reiter nach Ariadne2D	v	nicht umgesetzt, da Hilfen als nicht modale Dialoge umgesetzt wurden

Die gelisteten Findings wurden als Vorschlag gewertet, wovon zwei bereits umgesetzt wurden. Alle weiteren Vorschläge könnten in weiteren Versionen umgesetzt werden. Ausnahme bildet der Vorschlag zur Platzierung des Hilfebuttons. Hilfestellungen wurden beabsichtigt als nicht modale Dialoge in der Software realisiert, da diese dann zu jederzeit, ohne weitere Nutzeraktionen, sichtbar sind und verwendet werden können. Als positives Fazit wurde von den Teilnehmern die Arbeit mit der Strg-Taste gewertet.

8.2.6.6 *Individualisierbarkeit*

„Ein Dialog ist individualisierbar, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe, individuelle Vorlieben des Benutzers und Benutzerfähigkeiten zulässt“ (DIN Deutsches Institut für Normung e. V., 2006). Zu dieser Kategorie wurden keine Codes vergeben. Es können keine Aussagen dazu abgeleitet werden.

8.2.6.7 *Lernförderlichkeit*

„Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet“ (DIN Deutsches Institut für Normung e. V., 2006). Zu dieser Kategorie wurden keine Codes vergeben. Es können keine Aussagen dazu abgeleitet werden.

8.3 Iteration 2: Begutachtung des Prototypen durch Industrieexperten

Für die erste Iteration der Evaluation des Prototypen, in dem es nicht Ziel war dediziert einen Scaffold zu evaluieren, wurde eine möglichst konträre Nutzergruppe zu Studierenden gewählt, die möglichst viel Erfahrung in der Modellierung und Werkzeugen zur Modellierung haben.

8.3.1 *Ziele*

Ziel dieses Experiments war es herauszufinden, analog zur ersten Iteration, ob die Editoren, die Ariadne zur Verfügung stellt, von Experten so eingeschätzt werden,...

- dass alle gesuchten Elemente intuitiv auffindbar sind.
- dass alle gesuchten Elemente intuitiv einsetzbar sind.
- dass das [Diagramm][58] angelegt und gelöscht werden kann.
- dass das [Diagramm] gespeichert werden kann.
- dass Elementzusammenhänge funktionieren (Elemente innerhalb eines Diagramms sollten intuitiv verknüpft werden können).

Insgesamt sollten die oben aufgeworfenen Fragen zu Iteration 2 beantwortet werden:

- Wie schätzen Modellierungsexperten Benutzbarkeit und Einsetzbarkeit von Ariadne für Lehrveranstaltungen ein?
- Wie wird der Funktionsumfang von Ariadne von Modellierungsexperten beurteilt?

8.3.2 *Design*

Das Design der Studie ist als Begutachtung durch Experten angelegt. Experten werden schrittweise durch den Prototypen geleitet und zu ihrer Einschätzung befragt.

8.3.3 *Stichprobe*

Als Stichprobe sind ca. *N=5-10* Experten einer Konferenz zur Modellierung (Ende 2019) geplant, die Experten auf dem Gebiet der Modellierung von Softwaresystemen repräsentieren. Die veranschlagte Stichprobe ergibt sich aus der Aussage von Nielsen und Landauer (1993), die empirisch belegten, dass bereits fünf Experten in der Lage sind 85% der Usability-Probleme in einer Software zu finden.

8.3.4 *Instrumente*

Folgende Instrumente kommen zum Einsatz:

- Ariadne zum Modellieren der Diagramme

58 *Diagramm* stellt eine Variable dar, die mit dem je zu modellierenden Diagrammtyp ersetzt wird.

- Vorgefertige Diagramme, die bei Bedarf modelliert werden können
- Leitfragen (Teil 1):
  - Fehlen wesentliche Elemente in der Toolbox?
  - Was suchen Sie gerade?
  - Kommen Sie zurecht?
  - Würden Sie ein Element an einem anderen Ort erwarten?
  - Fehlt Ihnen etwas/ein Element?
  - Funktioniert der Editor intuitiv?
  - Funktionieren die Buttons wie erwartet?
  - Ist der Diagrammbaum sinnvoll und verständlich aufgebaut?
- Leitfragen (Teil 2):
  - Ist das Tool Ihrer Meinung nach einfach handhabbar?
  - Was erwarten Sie hinter dem Button Literatur?
  - Was erwarten Sie hinter dem Button Vorstellungshilfe?
  - Was erwarten Sie hinter dem Button Analysehilfe?
  - Was erwarten Sie hinter dem Button Anleitungen?
  - Allgemeines Feedback/Verbesserungsvorschläge
  - Würden Sie den Editor, wenn er frei verfügbar wäre, für eine Basismodellierung selbst einsetzen?
  - Halten Sie das Tool für geeignet um Anfängern die Modellierung mit der UML beizubringen und trotzdem für professionell und umfangreich genug?(Hinweis: Studierende sollen die UML lernen und deshalb an möglichst viele Details selbst denken)
- Audioaufnahme
- Screencast des Bildschirms

8.3.5 *Durchführung*

1. Aufklärung, dass es darum geht, einen intuitiv bedienbaren UML Editor anzubieten, der Studierenden die Modellierung mit der UML einfach macht, nicht zu komplex bedienbar ist und ihnen Hilfsmittel zur Verfügung stellt.
2. Die Teilnehmer werden gebeten ein Diagramm auszuwählen, den entsprechenden Editor dazu anzulegen und exemplarisch Elemente des Diagramms zu modellieren (optional auf Basis der vorgefertigten Diagramme). Dabei tritt der Experimentleitende[59] unter Nutzung der Leitfragen in den Diskurs mit dem Probanden und bittet

59 Der Experimentleitender ist in diesem Fall der Autor dieser Arbeit.

den Probanden Schritte, die er/sie vorhat bzw. gerade durchführen möchte, zu erläutern.
3. Der Experimentleitende provoziert einen Fehlerdialog in einem Diagramm und bittet Experten um Einschätzung (z. B. Ist die Positionierung der empfohlenen Hilfsmittel geeignet?)
4. Während der Modelliertätigkeiten (ab Schritt 2) werden Audioaufnahmen sowie ein Screencast der Modellierungsumgebung erstellt.

8.3.6 *Auswertung & Interpretation der Ergebnisse*

Die Auswertung dieser Iteration dient, wie auch die erste Iteration, der Verbesserung des Prototypen, mit dem Fokus auf die Baumstruktur als primäre Navigation und der Editoren zur Modellierung. Das Material wird mit Hilfe des gleichen Verfahrens der ersten Iteration ausgewertet. *N=6* Experten haben die Begutachtung durchgeführt. Da die Evaluation zur Verbesserung des Prototypen dient und als formative Evaluation klassifiziert wurde, scheint diese Größe der Stichprobe ausreichend.

8.3.6.1 *Aufgabenangemessenheit*

Laut der DIN Deutsches Institut für Normung e. V. (2006) ist ein Dialog „aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen." Auch die eingangs gestellte Frage, wie Experten den Prototypen Ariadne zum Einsatz in Lehrveranstaltungen einschätzen, zielt in eine ähnliche Richtung, wie die Beschreibung zur Dialoggestaltung. Tabelle 8.4 zeigt die Findings der Experten:

- Das Anlegen eines neuen Diagramms in der Baumstruktur wird als problematisch eingeschätzt. Dies deckt sich mit der Auswertung der ersten Iteration, in der Studierende Schwierigkeiten damit aufwiesen.
- Einhergehend mit dem erstgenannten Item wurde geäußert, dass der Aufbau der Baumstruktur zunächst nicht intuitiv erscheint. Nach einigen Erläuterungen konnte die Struktur nachvollzogen werden. Auf der Ebene des Klassendiagramms, ist aber deshalb nun für eine weitere Iteration das Hinzufügen von Aktivitätsdiagrammen hinzugefügt worden.
- Die Verlinkung einzelner Elemente und gesamter Diagrammarten (Requirement mit einer Aktion) wurden vorgeschlagen, wurden aber aus Sicht des Autors beabsichtigt nicht umgesetzt um Komplexität für Studierende zu reduzieren.

- Experten erwarten auf Basis eines Aktivitätsdiagramms die Möglichkeit der Codegenerierung. Dies wurde nicht umgesetzt, da Codegenerierung für den Tooleinsatz nicht im Fokus steht.
- Der Vorschlag die Klasse konfigurierbar zu gestalten um flexibel Stereotypen hinzufügen zu können, wurde zum Teil umgesetzt, in dem weitere Elemente mit passenden Stereotypen angeboten werden („abstract“, „enum“, „interface“)
- Alle weiteren Anmerkungen, deren Felder keine Beschreibung enthalten, können in weiteren Iterationen umgesetzt werden.

Tabelle 8.4: Findings aus der Expertenbegutachtung zum Usability Grundsatz der Aufgabenangemessenheit

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Diagramm anlegen im Baumstruktur	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
Aktivitätsdiagramm unterhalb des Klassendiagramms	v	umgesetzt
Möglichkeit der Codegenerierung aus Aktivitätsdiagramm	v	keine Problemlösung, Codegenerierung stand nicht im Fokus für Studierende
Verknüpfung von Elementen verschiedener Diagrammtypen	v	keine Lösung
Verlinkung von Aktion mit Requirement	v	keine Lösung
Abhängigkeiten zwischen Diagrammen nicht vorhanden	v	keine Lösung
Da es 12 verschiedene Stereotypen gibt, ist Klasse konfigurierbar?	v	Angebot weiterer Stereotypen
create und destroy Nachricht fehlen im Sequenzidagramm	x	umgesetzt

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Send und Receive Events im Aktivitätsdiagramm	v	
Definition Entry Action im Zustand	v	
parallele Zustände	v	

Als Fazit wurde die Bewertung der Nutzerführung durch die Experten als „nicht wirklich schwierig“ bewertet.

8.3.6.2 *Selbstbeschreibungsfähigkeit*

Tabelle 8.5 beschreibt die Findings, d. h. Einschätzungen der Experten, die der Kategorie der Selbstbeschreibungsfähigkeit zugeordnet wurden.

Tabelle 8.5: Findings aus der Expertenbegutachtung zum Usability Grundsatz der Selbstbeschreibungsfähigkeit

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Element System unklar (Use-Case-Diagramm)	x	
Baumstruktur nicht selbsterklärend	x	
TN findet Transitionsbeschreibung nicht	x	Eigenschaften über Drop-Down Menü erreichbar
TN findet Funktion zum Hinzufügen eines Editors nicht, Suche im Reiter	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
Annotationen direkt mit den Fehlern direkt an die Elemente	v	
Schriftart grau auf blau	x	Theme Support

Im Folgenden werden die zentralen Findings und Vorschläge der Teilnehmer interpretiert und Umsetzungen erläutert:

- Das Element „System“ wird bereitgestellt, um Studierenden die Möglichkeit zu bieten, das zu entwickelnde System zu modellieren

und darin Anwendungsfälle zu platzieren. Akteure werden in der Regel außerhalb des Systems platziert, was mit diesem Element gelingt.
- Ein Teilnehmer hat ebenso, wie bereits in der ersten Iteration das Feld zur Beschreibung einer Transition im Zustandsdiagramm nicht gefunden. Deshalb wurde der Zugang zum Dialog der Eigenschaften, in dem dieses Feld gefüllt werden kann, zusätzlich im Drop-Down Menü zu der Transition hinterlegt.
- Das Hinzufügen eines neuen Editors wird zunächst nicht gefunden und in einem der Reiter vermutet. Diese Problematik wurde bereits erläutert. Eine zusätzliche Möglichkeit wurde oberhalb der Baumstruktur geschaffen.
- Kritisiert wurde außerdem die Farbwahl des Designs. Dieses Problem wurde gelöst, in dem der Theme Support aus DevExpress integriert wurde.
- Nach der Vorstellung der Problemerkennung wurde der Vorschlag gemacht, die Problemstellen direkt an den Elementen zu vermerken. Dies wurde über die Färbung und Auswahl der betroffenen Elemente pro Problem zum Teil umgesetzt.

8.3.6.3 *Erwartungskonformität*

Neben dem Grundsatz der Aufgabenangemessenheit sind für die zugrundeliegende Fragestellung die Einschätzung der Experten zum Grundsatz der Erwartungskonformität interessant. Es werden alle Punkte (siehe Tabelle 8.6) gelistet, die Experten erwarten, aber nicht, oder auf anderem Weg erhalten:

- Das Hinzufügen eines neuen Diagramm-Editors stellt auch hier wieder ein Finding dar und wurde bereits erläutert. Die Baumstruktur ist erst nach Erläuterung nachvollziehbar. Gewünscht wird, einen Aktivitätsdiagramm-Editor auf Ebene des Klassendiagramm-Editors hinzufügen zu können.
- Zudem wurden wiederum weitere Versuche ohne Drag-and-Drop unternommen: z. B. sollte ein Verbinder mittels der Markierung von zuerst Quelle und anschließend Ziel erzeugt werden, weshalb der Verbindermodus integriert wurde. Weiterhin wurde versucht ein Element anzuwählen und es anschließend per Klick auf die Modellierungsfläche zu bringen. Nachdem die Drag-and-Drop-Funktionalität erläutert wurde, war diese allerdings allen Experten eingängig.
- In drei Diagrammarten wurden Elemente genannt, die der Toolbox noch hinzugefügt werden könnten (leere Felder). Für das Sequenz-

diagramm wurde bereits eine create- und destroy- Nachricht hinzugefügt. Im Zustandsdiagramm sollte die Entry-Action definiert werden könne, im Aktivitätsdiagramm werden Send- und Receive Elemente vermisst.
- Weiterhin fehlt Experten die Verknüpfbarkeit bzw. die Möglichkeit Elemente oder Diagramme zu verlinken, dies würden sie von einem Tool erwarten. Auf Nachfrage, ob dies auch für den Lehreinsatz unbedingt erforderlich sei, bestätigen sie diese Anforderung jedoch nicht.
- Für den Sequenzdiagramm-Editor wurden Verbesserungsvorschläge geliefert, so soll die Lebenslinie nicht explizit hinzugefügt werden müssen, der Aktivitätsbalken ausschließlich dynamisch nach unten wachsen können und Nachrichten gleich in der gängigen Verwendungsrichtung angezeigt werden.

Tabelle 8.6: Findings aus der Expertenbegutachtung zum Usability Grundsatz der Erwartungskonformität

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Findet Editor nicht, Suche im Reiter	x	zusätzliche Möglichkeit zum Einfügen oberhalb des Baums
Aktivitätsdiagramm unterhalb vom Klassendiagramm	v	umgesetzt
Verbinder mit Quelle Ziel Markierung	x	Verbindermodus
Verbindungsversuch ohne Drag-and-Drop	x	Verbindermodus
Auswahl eines Elements der Toolbox als ständige Auswahl	x	nicht umgesetzt
Element durch Anwahl und Klick auf Zeichenfläche	x	nicht umgesetzt (Alternative Funktion vorhanden)
Möglichkeit der Codegenerierung aus Aktivitätsdiagramm	v	keine Lösung, Codegenerierung stand nicht im Fokus für Studierende

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Möglichkeit zum „Losschreiben“ sobald Element markiert ist (ohne Doppelklick)	v	
Definition Entry Action im Zustand (Zustandsdiagramm)	v	
Send und Receive Events (Aktivitätsdiagramm)	v	
create und destroy fehlen (Sequenzdiagramm)	x	umgesetzt
Lebenslinie nicht extra hinzufügen	v	
Pfeil umdrehen, weil meist von Links nach Rechts gearbeitet wird, Antwortpfeil dementsprechend von Rechts nach Links	v	umgesetzt
Aktivitätsbalken sollte einfach nur nach unten wachsen, sonst verschieben sich die Nachrichten mit beim Skalieren (Sequenzdiagramm)	v	
Inhalt bei mehreren Zuständen verstecken (in Unterzustand)	v	
Verknüpfung von Elementen verschiedener Diagrammtypen	v	keine Lösung
Wiederverwendung von Klassen in verschiedenen Diagrammen	v	keine Lösung
Klassen bzw. Elemente verknüpfbar machen	v	keine Lösung

Code	Problem (x)/ Vorschlag (v)	Beschreibung der Lösungsstrategie
Verlinkung von Aktion mit Requirement	v	keine Lösung

Für die Grundsätze Fehlertoleranz, Steuerbarkeit sowie Individualisierbarkeit und Lernförderlichkeit wurden keine Findings identifiziert.

8.4 INTERPRETATION DER ERGEBNISSE & FAZIT

Insgesamt lagen den beiden Iterationen vier Fragen zu Grunde:

- Wie schätzen Studierende die Benutzbarkeit von Ariadne ein?
- Wie werden die Selbsterklärungsfähigkeit der Bestandteile, explizit der Scaffolds und deren Verwendung von Studierenden eingeschätzt?
- Wie schätzen Modellierungsexperten Benutzbarkeit und Einsetzbarkeit von Ariadne in Lehrveranstaltungen ein?
- Wie wird der Funktionsumfang von Ariadne von Modellierungsexperten beurteilt?

Die Interpretation der Erkenntnisse aus der Anwendung der Grundsätze zur Dialoggestaltung lassen folgende Findings zu:

- Das Hinzufügen neuer Diagramme war für beide Nutzergruppen schwierig, weshalb eine alternative Möglichkeit zum Hinzufügen der direkt für Nutzer sichtbaren Funktion oberhalb der Baumstruktur eingeführt wurde.
- Die Verbindung von Elementen wurde durch den Verbindermodus ergänzt.
- Der Zugang zum Dialog der Eigenschaften wurde ebenfalls durch eine alternative Funktion über den Rechtsklick auf eine Assoziation ergänzt.
- Die Problematik unbekannter Elemente unter der Zielgruppe der Studierenden wurde durch die default-Einstellung der Beschriftungsanzeige gelöst.

Insgesamt entstand über beide Iterationen hinweg der Eindruck, dass nach den Verbesserungen der Prototyp in seiner Benutzbarkeit grundsätzlich positiv eingeschätzt wurde. Bezogen auf die Selbsterklärungsfähigkeit lässt sich festhalten, dass die unbekannten Hilfestellungen nicht selbsterklärungsfähig sind, die bekannten Hilfestellungen wurden

entsprechend der Vorschläge angepasst und sollten demnach nun als selbstbeschreibungsfähig gelten. Modellierungsexperten stimmten mit der Autorin der Arbeit überein und bestätigten in der Diskussion einen möglichen Lehreinsatz, erwarten für einen täglichen Einsatz allerdings deutlich mehr Automatismen durch das Tool, wie beispielsweise die Vorgabe von Datentypen und Rückgabewerten sowie eine Syntaxvorgabe bei der Beschriftung von Transitionen. Insgesamt enthält der Prototyp bis auf die genannten Elemente alle standardmäßig notwendigen Elemente aus Sicht der Experten.

In den Evaluationen wurden von Teilnehmern Vergleiche gezogen. Diese wurden in den Auswertungen exkludiert. Ein Vergleich mit am Markt verfügbaren Werkzeuge soll im Rahmen dieser Arbeit hier nicht weiter verfolgt werden. Am Markt verfügbare Tools variieren sehr stark in Art und Umfang ihrer Funktionalität: Einige fokussieren z. B. die Simulation und automatisierte Generierung von Quellcode sowie die Darstellung möglichst komplexer Diagramme. In dieser Arbeit wurde die prototypisch Integration von Hilfestellungen intendiert, die so bislang in keinem der Tools angeboten werden. Eine Neuimplementierung wurde einer Erweiterung vorgezogen,...

- da eine möglichst einfache, intuitive Benutzerführung gestaltet werden sollte.
- da eine Unterstützung mit AR evaluiert werden sollte.
- da komplexe Hilfestellungen integraler Bestandteil der Software sein sollten.
- da die Verwendung der Hilfsmittel und der Software gelogged werden sollte.
- da Probleme der Studierenden detektiert werden sollten.

Keine der bekannten Tools, bot geeignete Möglichkeiten oder umfassende Schnittstellen im notwendigen Umfang für die Realisierung dieser Vorhaben (z. B. die Anpassung der Benutzerführung und der Oberfläche).

Die Mikro-Evaluation bzw. Begutachtung durch Experten zeigt heterogene Wünsche und Bedürfnisse auf. Experten schätzen den Prototyp als geeignet für den Lehreinsatz ein, wünschen sich für den Arbeitsalltag allerdings mehr Automatismen und weniger einschränkendes Feedback durch die Software. Studierende hingegen bewerten das Feedback der Software als positiv und empfinden die nicht verfügbaren Automatismen mehr als Freiheit in der Bedienung, weniger als fehlendes Feature.

Insgesamt war die Software in vier verschiedenen Iterationen im Einsatz. Zeitlich betrachtet ist der erste Einsatz von Ariadne im Wintersemester 2017/2018 im Experiment, das in Abschnitt 6.10.1 beschrieben ist, erfolgt. Anschließend folgen die beiden Iterationen mit Studierenden

und Experten, die in diesem Kapitel beschrieben wurden, der aktuelle Prototyp wurde im Sommersemester 2020 im Experiment zu EMMEs (Abschnitt 6.11) eingesetzt. Alle diese Evaluationen sind als formative Evaluationen zu bewerten. Zudem könnten weitere Evaluationen, die nicht die Usability als Fokus tragen, die Verwendung der verschiedenen Hilfsmittel betrachten. Mit Hilfe summativer Betrachtungen könnten auch Rückschlüsse auf die Effektivität des Tools getroffen werden.

Bezogen auf den vorgestellten Ansatz der formativen Evaluation nach van den Akker et al. (2010) und dem zugrunde gelegten Kriterium der Praxistauglichkeit, können erste Ergebnisse zur erfüllten Praxistauglichkeit abgeleitet werden.

9

ZUSAMMENFASSUNG DER ERGEBNISSE DER ARBEIT & AUSBLICK

Wer hohe Türme bauen will,
muß lange beim Fundament verweilen.

— Josef Anton Bruckner

Die bisherigen Ergebnisse und Umsetzung des Prototypen sind Basis und „Enabler" für weitere Forschung. Dieses abschließende Kapitel fasst die Ergebnisse der Arbeit (siehe Abschnitt 9.1) und gibt einen Ausblick sowie Ideen für weitere Forschung auf diesem Gebiet (Abschnitt 9.2).

9.1 ZUSAMMENFASSUNG DER WICHTIGSTEN ERGEBNISSE DER ARBEIT

Die Arbeit widmete sich der problembasierten Initiierung von Hilfsmitteln (Scaffolds) zur Unterstützung Studierender bei der Modellierung mit der Unified Modeling Language (UML). Als zentrales Artefakt wurde Ariadne als evidenzbasierte Modellierungsumgebung geschaffen, die die Modellierung von Softwaresystemen im Rahmen der objektorientierten Analyse mit verschiedenen integrierten Hilfsmitteln (sog. Scaffolds) unterstützt. Ausgangsbasis für die Umgebung sind katalogisierte Probleme bei der Modellierung von Softwaresystemen mit der UML. Um die identifizierten Probleme zu adressieren, wurden verschiedene Scaffolds entwickelt und evaluiert. Dazu gehören verschiedene Arten von Unterlagen und Tutorials als klassische Hilfsmittel, welche von Studierenden während des Modellierprozesses innerhalb der Umgebung von Ariadne abgerufen werden können. Ariadne beinhaltet, abgesehen von klassischen Hilfsmitteln, auch experimentelle Ansätze, wie die Verwendung von Augmented Reality, Eyetracking und Natural Language Processing (NLP) als Mittel zur Textanalyse.

Im folgenden werden nun zentrale Ergebnisse der einzelnen Kapitel vorgestellt.

9.1.1 *Theoretische Einordnung*

Kapitel drei und vier verorteten die Arbeit in ihrem interdisziplinärem Feld. Kapitel drei zeigte relevante pädagogische Theorien für Ist- und Sollzustand für die Verbesserung der Lehre zur Modellierung eines Softwaresystems auf: Als Ausgangslage gilt es Bedürfnisse und Probleme einer heterogenen Studierendengruppe zu kennen, zu erkennen und zu adressieren. Studierende weisen individuelle Konstitutionen von Lernhindernissen und Kognition auf. Als Soll-Zustand wird ein Ansatz zu einer konstruktivistischen Lernumgebung vorgeschlagen, der möglichst individuell, technologiebasierte Hilfestellungen zur Modellierung von Softwaresystemen bereitstellt.

Kapitel vier zeigt einen Abriss über das Forschungsfeld, die Rolle der UML im Software Engineering, genauer der 4+1 Sicht, im Softwarelebenszyklus und speziell in der Objektorientierung und schließt mit Betrachtung zur Lehre der UML im Hochschulkontext.

9.1.2 *Probleme Studierender bei der Modellierung mit der UML*

Das fünfte Kapitel zeigte eine Analyse der Probleme Studierender bei der Modellierung mit der UML. Ziel der Studie war es, den Problemkontext zu erfassen und existierende Probleme bzw. Schwierigkeiten Studierender bei der Modellierung mit der UML zu erfassen und zu katalogisieren, um anschließend Maßnahmen für eine computergestützte Lernunterstützung abzuleiten.

Existierende Probleme Studierender wurden mittels einer systematischen Literaturrecherche exzerpiert. Zusätzlich dazu wurden Studierende im Rahmen einer Fragebogenstudie nach aktuellen Schwierigkeiten befragt und Probleme Studierender während des Modellierungsprozesses mit Hilfe einer Beobachtungsstudie erhoben. Probleme, die in der Literatur genannt wurden sowie die Schwierigkeiten, die seitens Studierender in der Fragebogenstudie genannt wurden, wurden für ein initiales Codesystem verwendet, um die Beobachtungsstudie auszuwerten und Probleme während des Modellierungsprozesses zu katalogisieren. Auf Basis des Problemkataloges wurden Maßnahmen unter Berücksichtigung technologiebasierten Lernens abgeleitet. Damit wurde die erste Forschungsfrage nach den existierenden Problemen Studierender in der Modellierung beantwortet und durch Ableitung der Maßnahmen eine Übersicht für die Konzeption von Unterstützungsmaßnahmen geschaffen.

9.1.3 *Konzeption eigener Scaffolds*

Das sechste Kapitel präsentierte die konzeptionelle Erstellung der Scaffolds. Realisiert wurden mögliche Unterstützungsmaßnahmen, aufgeteilt in klassische Hilfsmittel und experimentelle technologiebasierte Hilfsmittel sowie Anforderungen an eine neu konzipierte Modellierungsumgebung für Studierende. Für den Einsatz von AR und EMMEs werden zudem Evaluationsergebnisse gezeigt.

Vorgestellte klassische Hilfsmittel:

- Unterlagen
- Tutorials
- Versionierung
- Kommentare

Vorgestellte technologiebasierte Hilfsmittel:

- Feedback
- Analysehilfen
- der Einsatz von AR
- der Einsatz von EMMEs

Die Konzeption der Hilfsmittel wurde zudem theoretisch verortet in der Theorie des Scaffolding. Das Feld der existierenden Scaffolds im Bereich Software Engineering wurde aufgezeigt.

In diesem Kapitel wird zudem die zweite Forschungsfrage nach möglichen Unterstützungsmöglichkeiten für Studierende für den Modellierungsprozess beantwortet.

9.1.4 *Ariadne*

Im siebten Kapitel lag das Hauptaugenmerk auf der selbst entwickelten Modellierungsumgebung Ariadne. Beschrieben wurden die grundlegende Systemarchitektur, die drei verwendeten Datenbanken sowie das zugrundeliegende Rollenkonzept. Außerdem wurden relevante Komponenten bzw. Konzepte betrachtet:

- der schematische Aufbau der einzelnen Editoren
- der Zusammenhang zwischen Scaffolds und detektierten Problemen mit Hilfe der Recommendation-Datenbank
- die Möglichkeit zur Bewertung von Scaffolds durch Dozierende und Studierende
- der gewählte Ansatz zur Detektion der Probleme

9.1.5 *Rechtfertigung & Evaluation*

Das achte Kapitel beschreibt zwei Iterationen mit der Modellierungsumgebung, eine Mikro-Evaluation mit Studierenden sowie eine Begutachtung durch Industrieexperten, mit dem Ziel, das allgemeine Konzept von Ariadne mit Hilfe möglichst konträrer Nutzergruppen beurteilen zu lassen. In den beiden Studien stand die Modellierungsumgebung Ariadne im Vordergrund. Einzelne Evaluationen von Hilfsmitteln wurden in Kapitel sechs bereits vorgestellt.

9.2 AUSBLICK & MÖGLICHE WEITERE ARBEITEN

Nach der zusammenfassenden Darstellung der relevanten Beiträge der Arbeit, werden im Folgenden einige Möglichkeiten Weiterentwicklung und zu weiteren Forschungsarbeiten vorgestellt.

9.2.1 *Expertenmodus*

Die zweite Iteration der Evaluation, in der Experten die Struktur und die Modelleditoren von Ariadne begutachtet haben, hat deutliche Bedarfsdifferenzen aufgezeigt. Bedürfnisse von Experten, wenn sie Ariadne im täglichen Arbeitseinsatz hätten, unterscheiden sich von dem derzeitigen Stand des Prototypen, als dass Experten mehr Automatismen bzw. Funktionen zur Autovervollständigung erwarten.

Um für beide Nutzergruppen positive User Experience (DIN Deutsches Institut für Normung e. V. (2006)) zu ermöglichen, wurde ein Expertenmodus vorgesehen, in dem beispielsweise Transitionen in Zustandsautomaten bereits eine Syntax vorgeben, ebenso Definitionen von Methoden und Attributen im Klassendiagramm.

9.2.2 *Optimierung der Scaffolds & Implementierung weiterer Scaffolds*

Im Rahmen der Konzeption der Analysehilfen wurden Verfahren zur Textanalyse mit Hilfe von NLP (OpenNLP) vorgestellt. Derzeit werden hier frei verfügbare vorhandene Modelle verwendet. Um die Hilfestellungen zu verbessern, könnten beispielsweise eigene Trainingsmodelle für die Erkennung geeigneter Textstellen bzw. Phrasen für die Detektion von Use-Cases sowie Use-Case Benennungen erstellt werden. Auch die Substantiv-Verb-Analyse könnte damit verbessert werden und im Verlauf möglicherweise sogar eine Eignung von Klassen priorisieren.

Ebenso ist das Bewertungssystem mithilfe dessen Nutzer Hilfsmittel bewerten können, nur prototypisch realisiert und die grundlegende Funktionsfähigkeit sichergestellt. Eine Nutzung durch große Nutzerkreise würde hier reale Rankings der zur Verfügung stehenden Hilfsmittel ermöglichen. Ein Einsatz durch viele Anwender (Studierende und Dozierende) und eine Integration entsprechender Trainingsmodelle für die verschiedenen Verfahren zur Identifikation von Use-Cases, Klassen und zur Bewertung ermöglicht es, dass Ariadne ein intelligentes System wird, das Hilfestellungen empfehlen kann und intelligente Hilfestellungen bereitstellt.

Die Problemerkennung kann von Dozierenden flexibel erweitert und verändert werden. Um diese Funktion nutzen zu können, müssen Experten XSLT implementieren können, sowie die XML Struktur von Ariadne und das XMI Schema kennen. Auch hier könnte im Rahmen von Weiterentwicklungen ein weniger komplexes User Interface implementiert werden, es sollte aber ebenfalls durch die Detektion weiterer Problemstellen vervollständigt werden.

9.2.3 *Implementierung weiterer Editoren & Notationsvarianten*

Im Diskurs während der Begutachtung durch Experten wurden auch Rückfragen zur Auswahl der Editoren und zum Einsatz von SysML[60] gestellt. Die Auswahl der zur Verfügung stehenden Editoren liegt darin begründet, dass diese zu den am häufigsten verwendeten Diagrammtypen in Lehre und Praxis gelten (Anke & Bente, 2019; Frezza & Andersen, 2006; Granda et al., 2015). Eine Erweiterung (oder auch Einschränkung) durch weitere Diagrammtypen ist möglich. Die Realisierung der Editoren wurde in Abschnitt 7.2.3.2 gezeigt. Auf dieser Basis können die bestehenden Editoren für die SysML erweitert werden. Die SysML besteht aus einer Untermenge der in der UML 2 definierten Diagrammtypen. In Ariadne sind bereits vier Diagrammtypen, die auch in der SysML angewendet werden, realisiert (Use-Case-Diagramm, Aktivitätsdiagramm, Zustandsdiagramm und Sequenzdiagramm).

9.2.4 *Cloudanbindung*

In Abschnitt 7.1.2 wurden drei Datenbankmodelle vorgestellt. Für alle durchgeführten Experimente wurde ein SQL-Server verwendet, der im lokalen Netzwerk zur Verfügung stand. Um Ariadne außerhalb der Netzwerks verwenden zu können, müssen die Datenbanken *AriadneDB*

60 https://sysml.org/

und *AriadneRecommendationDB* beispielsweise in Microsoft Azure zur Verfügung gestellt werden. Für die *AriadneTelemetry*- Datenbank, die zum Logging verschiedener Nutzerinteraktionen zur Verfügung steht, ist dies, wenn Eyetracking-Daten geloggt werden sollen, aufgrund der unkalkulierbaren Größe und dem schnellen Wachstum allerdings nicht zu empfehlen. In Anbetracht der sehr individuellen Ausgestaltungsmöglichkeiten der *AriadneRecommendationDB* gilt die Empfehlung für diese Datenbank ebenfalls.

9.2.5 *Optimierung der Nutzungsanalysen für Lehrende*

Die Häufigkeit der Verwendung von Scaffolds und zugehörige Problemstellen stellen ebenfalls interessante Parameter für die Evaluation von Scaffolds dar. Durch die Integration des Logging Application Blocks (siehe Abschnitt 7.3.5) ist der Grundstein dafür bereits gelegt. Abbildung 7.32 zeigt einige Logging-Daten zur Nutzung der Hilfsmittel, die derzeit noch nicht ausgewertet werden. Eine Aufbereitung der Daten in Berichte und deren Integration in die Unterstützungsmaßnahmen können z. B. dazu beitragen, Lehrenden Nutzungsanalysen der von ihnen bereitgestellten Hilfsmittel anzuzeigen. Erweiterungen könnten auch Rückschlüsse zwischen Problemen der Studierenden und gewählten Hilfsmittel ermitteln und optimieren.

9.2.6 *Mögliche Forschungsfelder & weitere Forschungsarbeiten*

Im Bereich der technologiebasierten Scaffolds wurde experimentell der Einsatz von AR und von EMMEs evaluiert. Alle weiteren Hilfsmittel wurden nicht explizit evaluiert und sollten in weiteren Experimenten noch Begutachtungen oder weiteren formativen Evaluationen unterzogen werden. Zudem wurde noch keine summative Evaluation durchgeführt. Auch der Einsatz von AR und EMMEs wurde nur im Rahmen von Querschnittstudien betrachtet, weshalb auf Basis der bisherigen Ergebnisse keine verallgemeinernden Aussagen getroffen werden können.

Die derzeitige Implementierung von Ariadne mit C# und WPF lässt prinzipiell eine Portierung der gesamten Applikation auf die Microsoft Hololens zu. Damit eröffnet sich ein weiteres Forschungsfeld, in dem beispielsweise neue Interaktionsmöglichkeiten unter dem Einsatz von AR untersucht werden konnten.

Weiteres Potenzial wird durch die Integration des Logging Application Blocks vermutet. Durch diesen ist es möglich, wie bereits aufgezeigt (siehe Abschnitt 7.3.5), beispielsweise auch Daten zu loggen, die zur Validierung von Eyetracking-Daten dienen können. Derzeit können mit

Hilfe des Tobii Spectrum 600 Eyetracking-Daten laut des Herstellers im Optimalfall (Ausrichtung mit Kinnstütze, optimaler Belichtung, Mitte des Bildschirms und einem Abstand von 65 cm) mit einer Genauigkeit von 0.34 cm aufgenommen werden (Tobii (2018)). Tatsächlich kann diese aber sehr viel höher liegen (z. B. 0.7 cm nach eigener Erfahrung). Dies setzt eine Interpretation der Forscher voraus, da je nach Kalibrierung keine eindeutigen Ergebnisse erkannt werden können. Durch die Nutzung des Logging-Frameworks können Analysen an Qualität und Aussagekraft gewinnen, da semantische Bedeutungen softwareseitig bestimmt werden können und eine zusätzliche Möglichkeit geschaffen wird, Bildschirmkoordinaten zu loggen. Dem Forscher wird damit die Auswertung der Daten zudem erleichtert, da Daten beispielsweise nicht mehr im Nachhinein codiert werden müssen.

9.2.7 *Abschließende Betrachtung*

Durch diese Arbeit ergeben sich eine Reihe an möglichen Forschungsdesideraten, die noch betrachtet werden können. In den obigen Ausführungen wurden nur beispielhaft Potenziale aufgezeigt. Diese Arbeit betrachtet auf makroskopischer Ebene didaktische Implikationen für die Lehre im Fach Software Engineering, speziell für objektorientierte Analyse und Design mit der UML und stellt Erkenntnisse zusammen, die zeigen, wie eine unterstützende Vermittlung gestaltet sein kann.

Es zeigten sich im Zuge der Arbeit aber auch Forschungsfelder (z. B. AR, EMME, Textanalyse etc.), die für den Bereich der Verbesserung der Lehre von Software Engineering interessant erscheinen und ebenso andere Disziplinen und Bereiche der praktischen Hochschuldidaktik bereichern könnten. Die bisherigen Ergebnisse belegen, dass es von Relevanz erscheint, im Bereich der praktischen Hochschuldidaktik evidenzbasiert geeignete Lehr-Lernsettings zu schaffen.

A

ANHÄNGE

Inhaltsübersicht:

- Datenbankmodelle
- Startscreen
- Fragebogen Selbsteinschätzung zum AR Experiment „neue Interaktionsmöglichkeit zur Steigerung von Motivation und Lernerfolg“
- Hinweise Think-Aloud
- Beobachtungsbogen und Angaben zum Experiment „AR und Abstraktionsfähigkeit“
- Vorwissenserhebung zum Experiment mit Eye-Movement Modeling Examples

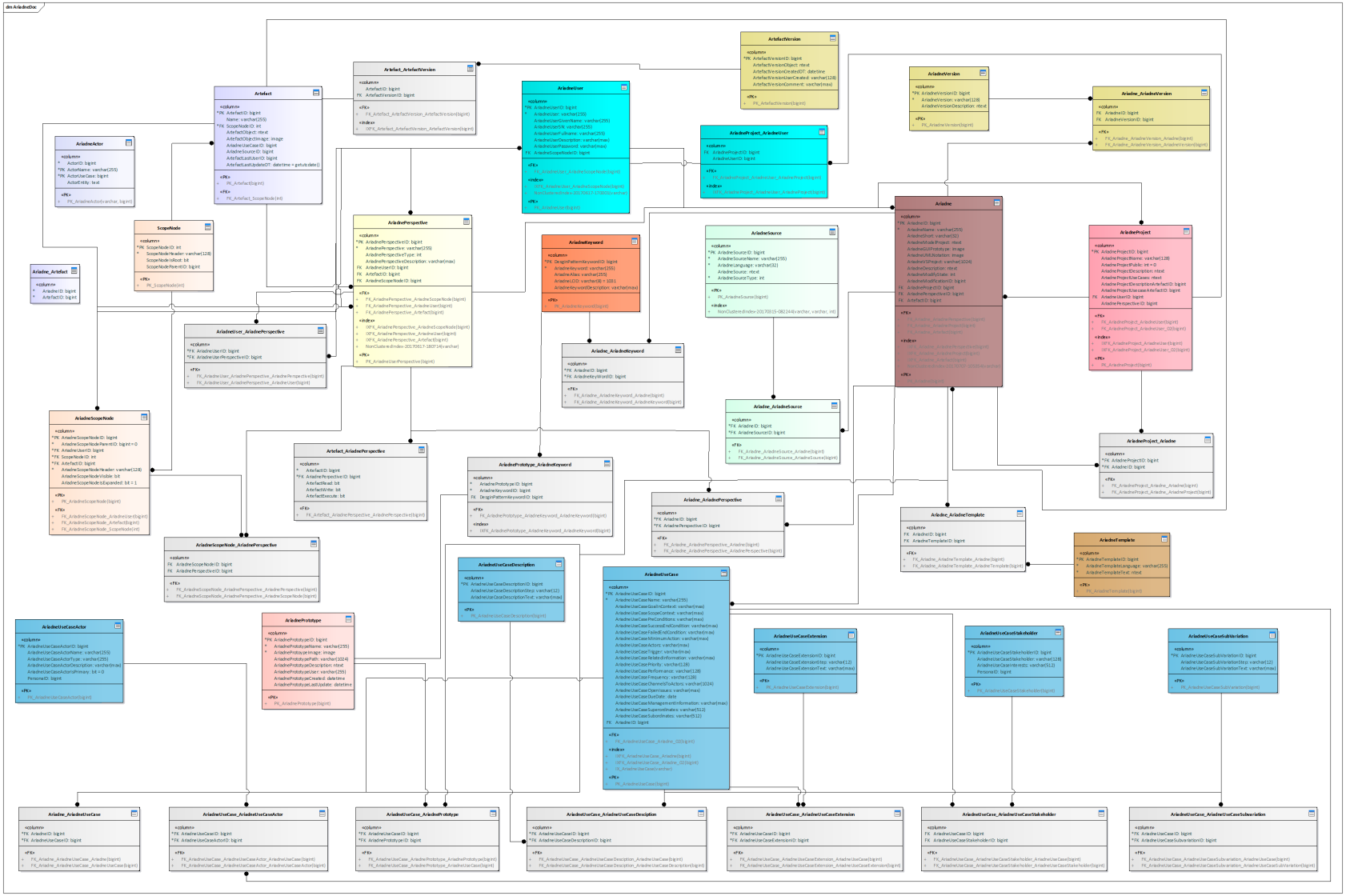


Abbildung A.1: Datenmodell der *Ariadne2D*-Datenbank

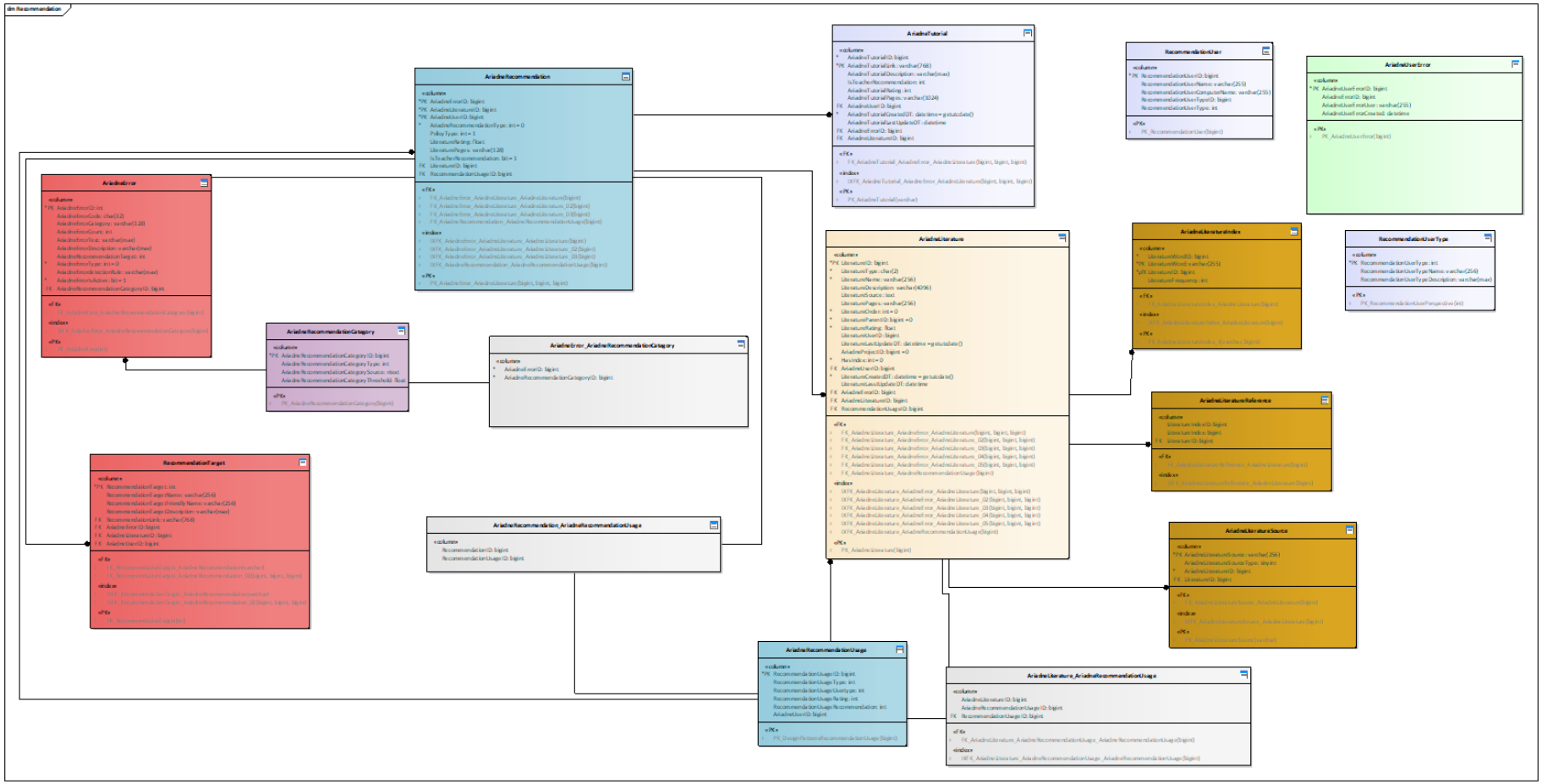


Abbildung A.2: Datenmodell der *AriadneRecommendation*-Datenbank

A.2 ANMELDEDIALOG

Abbildung A.3: Anmeldedialog zur Anmeldung bei Ariadne. Die NutzerInnen bestätigen das Logging nutzerspezifischer Daten.

A.3 Fragebogen zur Selbsteinschätzung zum AR Experiment

LaS [3]
Übung SES SoSe18

Der Fragebogen wird maschinell erfasst. Bitte mit Kugelschreiber oder nicht zu dickem Filzstift ausfüllen.
Ankreuzen: ☒ Bei Auswahlfeldern dürfen mehrere Antworten angekreuzt werden.
Korrigieren: ■ Bei Bewertungsfragen (Skala 1–6) darf nur ein Kästchen angekreuzt werden.

1 Allgemeines

Zur Erstellung eines Pseudonyms beantworten Sie bitte folgende Fragen zur Erstellung Ihres persönlichen Codes:

- Bitte geben Sie den ersten Buchstaben des Vornamens Ihrer Mutter (oder einer Person, die für Sie einer Mutter am nächsten kommt) an (z.B.: Anna-Marie = A).
- Bitte geben Sie Ihren Geburtsmonat in zwei Ziffern an (z.B.: März = 03).
- Bitte geben Sie den ersten Buchstaben des Vornamens Ihres Vaters (oder einer Person, die für Sie einem Vater am nächsten kommt) an (z.B.: Hans-Peter = H).
- Bitte geben Sie die Anzahl Ihrer Geschwister an (z.B. ein Bruder & eine Schwester = 2).
- Beispiel: A03H2

2 Bitte bewerten Sie die Fragen in diesem Abschnitt auf der folgenden Skala: Trifft gar nicht zu – Stimme stark zu.

2.1 Ich kann eine Klasse in UML modellieren.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.2 Ich kann verschiedene Beziehungen zwischen Klassen unterscheiden.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.3 Ich kann die Pfeilart einer Komposition identifizieren.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.4 Ich kann die Pfeilart einer Aggregation identifizieren.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.5 Ich kann die Pfeilart einer Vererbung identifizieren.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.6 Ich kenne den Unterschied zwischen einer Aggregation und einer Komposition.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.7 Ich kann eine Aggregationsverbindung korrekt einsetzen.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

NONE 3420821876 0001

2.8 Ich kann eine Kompositionsverbindung korrekt einsetzen.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.9 Ich weiß, wann eine Vererbungsbeziehung in UML verwendet wird.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.10 Ich weiß, wozu eine abstrakte Klasse verwendet wird.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.11 Ich weiß, wie eine abstrakte Klasse in UML modelliert wird.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.12 Ich weiß, wozu ein Interface verwendet wird.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

2.13 Ich weiß, wie ein Interface in UML modelliert wird.

☐ Trifft gar nicht zu ☐ Trifft nicht zu ☐ Trifft eher nicht zu ☐ Stimme eher zu ☐ Stimme zu ☐ Stimme stark zu

A.4 HINWEISE FÜR STUDIERENDE ZU „THINK-ALOUD“

THINK ALOUD— Denken Sie laut!

- Ich kann mir vorstellen...
- Eine Frage, die ich habe, ist...
- Das erinnert mich an...
- Das ist wie bei einem...
- Ich bin verwirrt über...
- Die große Idee hier ist...
- Ich glaube...

- „ ... jetzt überlege ich gerade wie ...“
- „ ... das Programm macht auf mich einen überladenen Eindruck ... “
- „ ... auf dem Bildschirm suche ich ...“
- „ ... nun wird es interessant ...“
- „ ... durch die Rückmeldung des Programms bin ich verunsichert ... “
- „ ... die Bedeutung der Buttons ist mir ein Rätsel ... “

A.5 BEOBACHTUNGSBOGEN/ ANGABEN ZUM AR EXPERIMENT

Beobachtungsbogen

Datum:

Testsubjekt ID:

Vorwissen:

Aufgabe 1:

Aufgabe 3:

Würden Sie weitere Dokumente nutzen um Ihre Erklärungen zu untermauern?

UML Diagramme: Sequenz-, Klassen-, Use-Case-, Zustandsautomat – welche würde man wo einsetzen?

Wie findest du die Bedienung der HoloLens und der App?

Hast du das Gefühl aus der App irgendwas gelernt zu haben?

Wie findest du die App im Vergleich mit der Übung?

Sonstige Anmerkungen?

Angaben

Angenommen Sie sind der Erfinder des ersten Kaffeeautomaten – das heißt niemand außer Ihnen kennt seine Funktionsweise. Der Kaffeeautomat hat alle unten aufgeführten Funktionen. Sie wollen nun die Software für den Kaffeeautomaten entwickeln lassen. ***Erklären Sie dem Entwickler die Funktionsweise des Kaffeeautomaten, sodass er die Software entwickeln kann.***

Nachdem der Kaffeeautomat fertig entwickelt ist, stellen Sie ihn auf einer Messe vor. Ein technisch unerfahrener Kunde kommt auf Sie zu und zeigt Interesse an Ihrem Kaffeeautomaten. ***Erklären Sie Ihrem Kunden die Funktionsweise des Kaffeeautomaten, sodass er ihn zuhause einsetzen kann.***

Informationen zum Kaffeeautomaten

Getränke

- Heißes Wasser
- Heiße Milch
- Schwarzer Kaffee
- Weißer Kaffee
- Cappuccino

Zu jedem Getränk hat die Maschine einen Knopf, mit dem die Zubereitung gestartet wird.

Bauteile

Alle Bauteile können über bestimmte Assembler Befehle gesteuert werden.

- Wassertank
  - Für Getränke, die Wasser benötigen, wird das Wasser hier eingegeben.
  - Falls der Wasserstand im Tank unter 200ml fällt, wird im Cache der Maschine eine Variable gesetzt, die besagt, dass nicht ausreichend Wasser für die Zubereitung von Getränken vorhanden ist.
- Milchtank
  - Für Getränke, die Milch benötigen, wird die Milch hier eingegeben.
- Wasserkocher
  - Um Schäden zu vermeiden verfügt der Wasserkocher über eine Schaltung, die die gesamte Kaffeemaschine abschaltet, wenn das Wasser über 110°C erhitzt wird.
- Milchkocher
  - Der Milchkocher verfügt über den gleichen Sicherheitsmechanismus wie der Wasserkocher.
- Milchaufschäumer
- Mahlwerk
  - Kaffeebohnen werden hier eingegeben.
- Filter
- Ausgabeeinheit
  - Diese hat für Wasser, Milch und Kaffee je eine separate Öffnung.
- Alle Bauteile sind mit Leitungen verbunden, die benötigt werden um die internen Prozesse zu realisieren.
  - Die Leitungen bestehen aus Edelstahl, um die Langlebigkeit der Maschine sicherzustellen.
- Display

    - Das Display zeigt immer den derzeitigen Zustand der Maschine an.
    - Das Display besteht aus einem 52x17 Raster von LEDs, die je einen Stromverbrauch von 8 Watt haben.

Interne Prozesse

Die beschriebenen Prozesse werden nur eingeleitet, falls alle nötigen Zutaten vorhanden sind. Auf dem Display wird eine Fehlermeldung angezeigt, falls nicht alle nötigen Zutaten vorhanden sind.

- Heißes Wasser
    - Wasser fließt vom Wassertank in den Wasserkocher.
    - Wasser wird im Wasserkocher erhitzt.
    - Heißes Wasser fließt vom Wasserkocher zur Ausgabeeinheit und wird ausgegeben.
- Heiße Milch
    - Milch fließt vom Milchtank in den Milchkocher
    - Milch wird im Milchkocher erhitzt.
    - Heiße Milch fließt vom Milchkocher zur Ausgabeeinheit und wird ausgegeben.
- Schwarzer Kaffee
    - Bohnen werden im Mahlwerk gemahlen.
    - Gemahlene Bohnen werden in den Filter geleitet.
    - Wasser wird gekocht und in den Filter geleitet.
    - Das Wasser wird durch den Filter geleitet und so zum Kaffee, der zur Ausgabeeinheit fließt und ausgegeben wird.
- Weißer Kaffee
    - Schwarzer Kaffee wird zubereitet.
    - Milch wird vom Milchtank zur Ausgabeeinheit geleitet und ausgegeben.
- Cappuccino
    - Schwarzer Kaffee wird zubereitet.
    - Milch wird gekocht und in den Milchaufschäumer geleitet.
    - Milch wird aufgeschäumt.
    - Aufgeschäumte Milch wird zur Ausgabeeinheit geleitet und ausgegeben.

Wartung

Bei regelmäßiger Benutzung muss das Mahlwerk und der Filter mindestens einmal die Woche gereinigt werden. Beide Bauteile sind herausnehmbar.

Zudem hat der Kaffeeautomat eine automatische Reinigungsfunktion, die jeden Tag ausgeführt werden sollte. Dazu muss der Kaffeeautomat leer sein.

Modelle

Der Kaffeeautomat ist in verschiedenen Farben verfügbar:

- Silber/Grau
- Schwarz
- Blau
- Pink
- Weiß

N = niemand

K = Kunde

E = Entwickler

Informationen zum Kaffeeautomaten

Getränke KE

- Heißes Wasser
- Heiße Milch
- Schwarzer Kaffee
- Weißer Kaffee
- Cappuccino

Zu jedem Getränk hat die Maschine einen Knopf, mit dem die Zubereitung gestartet wird.

Bauteile

Alle Bauteile können über bestimmte Assembler Befehle gesteuert werden. E

- Wassertank KE
  - Für Getränke, die Wasser benötigen, wird das Wasser hier eingegeben. KE
  - Falls der Wasserstand im Tank unter 200ml fällt, wird im Cache der Maschine eine Variable gesetzt, die besagt, dass nicht ausreichend Wasser für die Zubereitung von Getränken vorhanden ist. E
- Milchtank KE
  - Für Getränke, die Milch benötigen, wird die Milch hier eingegeben. KE
- Wasserkocher E
  - Um Schäden zu vermeiden verfügt der Wasserkocher über eine Schaltung, die die gesamte Kaffeemaschine abschaltet, wenn das Wasser über 110°C erhitzt wird. N
- Milchkocher
  - Der Milchkocher verfügt über den gleichen Sicherheitsmechanismus wie der Wasserkocher. N
- Milchaufschäumer E
- Mahlwerk KE
  - Kaffeebohnen werden hier eingegeben. KE
- Filter KE
- Ausgabeeinheit E
  - Diese hat für Wasser, Milch und Kaffee je eine separate Öffnung. N
- Alle Bauteile sind mit Leitungen verbunden, die benötigt werden um die internen Prozesse zu realisieren. E
  - Die Leitungen bestehen aus Edelstahl, um die Langlebigkeit der Maschine sicherzustellen. N
- Display
  - Das Display zeigt immer den derzeitigen Zustand der Maschine an. KE
  - Das Display besteht aus einem 52x17 Raster von LEDs, die je einen Stromverbrauch von 8 Watt haben. N

Interne Prozesse

Die beschriebenen Prozesse werden nur eingeleitet, falls alle nötigen Zutaten vorhanden sind. E Auf dem Display wird eine Fehlermeldung angezeigt, falls nicht alle nötigen Zutaten vorhanden sind. KE

- Heißes Wasser E
    - Wasser fließt vom Wassertank in den Wasserkocher.
    - Wasser wird im Wasserkocher erhitzt.
    - Heißes Wasser fließt vom Wasserkocher zur Ausgabeeinheit und wird ausgegeben.
- Heiße Milch E
    - Milch fließt vom Milchtank in den Milchkocher
    - Milch wird im Milchkocher erhitzt.
    - Heiße Milch fließt vom Milchkocher zur Ausgabeeinheit und wird ausgegeben.
- Schwarzer Kaffee E
    - Bohnen werden im Mahlwerk gemahlen.
    - Gemahlene Bohnen werden in den Filter geleitet.
    - Wasser wird gekocht und in den Filter geleitet.
    - Das Wasser wird durch den Filter geleitet und so zum Kaffee, der zur Ausgabeeinheit fließt und ausgegeben wird.
- Weißer Kaffee E
    - Schwarzer Kaffee wird zubereitet.
    - Milch wird vom Milchtank zur Ausgabeeinheit geleitet und ausgegeben.
- Cappuccino E
    - Schwarzer Kaffee wird zubereitet.
    - Milch wird gekocht und in den Milchaufschäumer geleitet.
    - Milch wird aufgeschäumt.
    - Aufgeschäumte Milch wird zur Ausgabeeinheit geleitet und ausgegeben.

Wartung

Bei regelmäßiger Benutzung muss das Mahlwerk und der Filter mindestens einmal die Woche gereinigt werden. Beide Bauteile sind herausnehmbar. K

Zudem hat der Kaffeeautomat eine automatische Reinigungsfunktion, die jeden Tag ausgeführt werden sollte. Dazu muss der Kaffeeautomat leer sein. KE

Modelle K

Der Kaffeeautomat ist in verschiedenen Farben verfügbar:

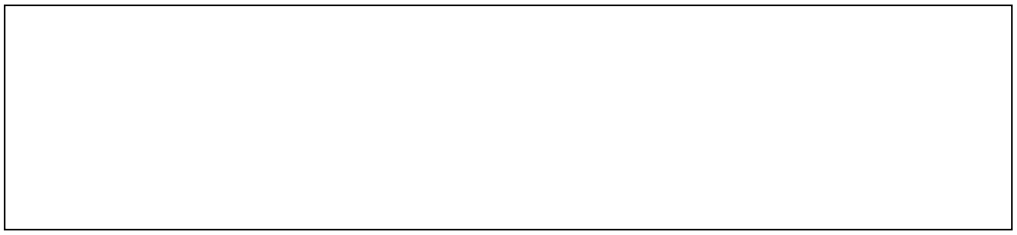
- Silber/Grau
- Schwarz
- Blau
- Pink
- Weiß

A.6 VORWISSENSERHEBUNG ZUM EXPERIMENT MIT EMMES

Teilnehmercode ____

Bitte bearbeiten Sie die nachfolgenden Aufgaben:

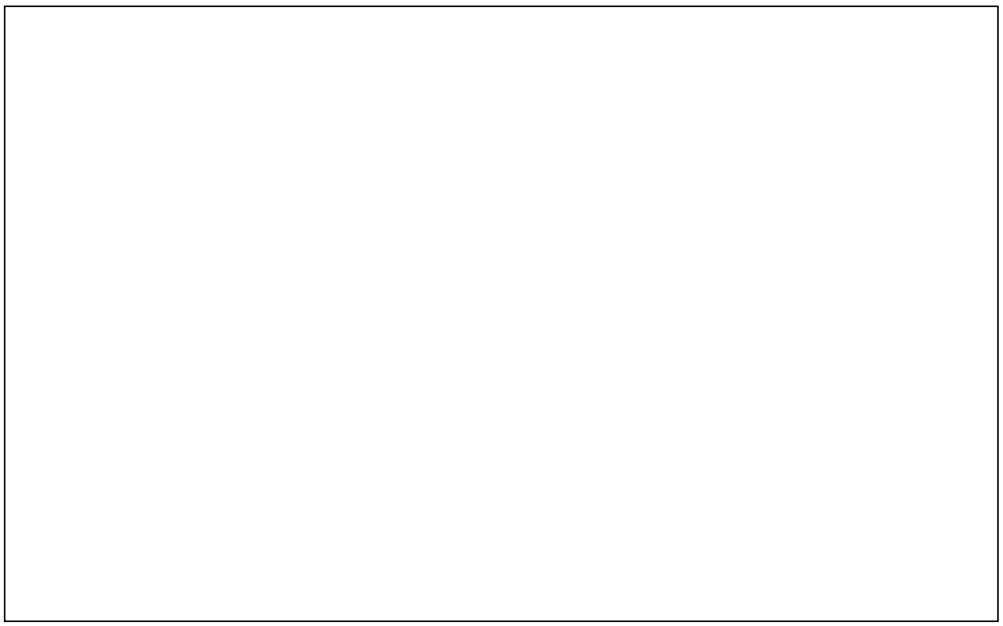
1) Modellieren Sie: Objekt A sendet Objekt B eine synchrone Nachricht und B bestätigt den Erhalt.



2) Stimmen Sie folgender Aussage zu?
In einem Sequenzdiagramm lässt sich eine zeitliche Ordnung feststellen.

(A) Ja
(B) Nein

3) Wie würden Sie folgendes Szenario modellieren? Die erste Nachricht von Objekt 1 erzeugt Objekt 3. Die zweite Nachricht (searchBook) ist eine synchrone Nachricht von Objekt 1 zu Objekt 3, die auf die Antwort von Objekt 3 (book (5)) wartet. Der Output-Parameter von Objekt 3 ist x=5. Nach der Suche wird die Lebenslinie angehalten. Objekt 1 und 2 tauschen das Signal takeOrder bzw. orderReceived aus (Objekt 1 ist der Sender des Signals.



Teilnehmercode ____

4) Stimmen Sie folgender Aussage zu?
Im Sequenzdiagramm darf es einen Aktor geben.

(A) Ja
(B) Nein

5) Stimmen Sie folgender Aussage zu?
Sequenzdiagramme bilden objektorientiertes Verhalten ab.
(A) Ja
(B) Nein

6) Modellieren Sie: Objekt A sendet Objekt B eine asynchrone Nachricht.

7) Modellieren Sie: Objekt A und Objekt B existieren bereits von Anfang an im Sequenzdiagramm, Objekt C wird von Objekt B dann initial erstellt.

8) Stimmen Sie folgender Aussage zu?
Bei asynchronen Nachrichten werden keine Antwortnachrichten gesendet.

(A) Ja
(B) Nein

A.7 BEWERTUNG DER SEQUENZDIAGRAMME IM EMME-EXPERIMENT

Statement	Rating	Value	Justification (optional)
Syntax (ground-level)			
There are no invalid elements • Elements that do not belong to sequence diagrams • Elements that miss a connection • Lost/found messages are not implemented in Ariadne	Yes	3	
Object names are compliant to the naming scheme • Scheme: [Name]:Class	Yes	3	
All actors are represented correctly • Actors must not be displayed as objects	Rather No	#NAME?	
Messages of stereotype <<create>> directly hit the object box • When an object is created, the message should directly hit the object box rather than its lifeline	Rather Yes	2	
The destruction of an object is shown by a cross on its lifeline	Yes	3	
All messages are labeled • Every message should be described by a string • The label of return messages can be omitted if it is obvious	Yes	3	
Every object posesses a lifeline	Yes	3	
The diagram is syntactically correct			of 21

Semantics (standing-level)			
The naming is consistent to the text • Object and/or class names match substantives in the text • Methods/operations match verbs in the text	Yes	3	
The naming is consistent to the class diagram • Object and/or class names match classes in the class diagram	Rather No	#NAME?	
The interaction is consistent to the class diagram • The interacting objects are connected via associations in the class diagram	Rather No	#NAME?	
The message sequence is consistent to the text	Yes	#NAME?	
The messages' meanings are consistent to their classes	Yes	3	
Objects are created consistent to the task • Objects are created/destroyed at the right time and place • I.e. they are created by the actors or objects that are meant to create them	Yes	3	
The diagram is consistent			of 18
There are no missing actors • At least the ones interacting with the system • For more problem space oriented diagrams, interaction between actors is possible	Yes	3	
There are no missing objects • The diagram contains at least the objects referred in the scenario	No	0	
There are no missing fragments • Branches in the scenario are reproduced correctly	Yes	3	
There are no missing messages • The diagram contains at least the operations referred in the scenario	Rather No	1	
There are no missing key elements • Explicitly mentioned behavior should not be abstracted away (i.e. contained in other elements)	Rather No	1	
The diagram is complete		8	of 15
The message labels/signatures are meaningful • Use appropriate message names instead of signatures in the problem space	Yes	3	
The diagram contains no elements with an unclear meaning	Yes	3	
The diagram is semantically correct		6	of 6

Statement	Rating	Value	Justification (optional)
Aesthetics (birds-eye view)			
The diagram is restricted to the scenario • Only the scenario is modeled, not the whole use case	Yes	3	
The diagram's level of detail is appropriate • Appropriate number of actors/objects/messages • Are there elements, which could be left out?	Rather Yes	2	
The diagram's level of detail is uniform • Does the diagram fit a single modeling space?	Yes	3	
The diagram is understandable by its intended users	Yes	3	
The message flow is oriented from the left to the right • Return messages are oriented the other way	Yes	3	
The diagram contains no redundant elements • Is the redundance necessary? • Can it be avoided with a fragment?	Yes	3	
The diagram is comprehensible		17	of 18

Overall rating		31	of 78

LITERATUR

Aamodt, A. & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.*, *7*(1), 39–59. doi:10.3233/aic-1994-7104

Abbott, R. J. (1983). Program Design by Informal English Descriptions. *Communications of the ACM*, *26*(11), 882–894. doi:10.1145/182.358441

Abke, J., Brune, P., Haupt, W., Hagel, G., Landes, D., Mottok, J., … Sedelmaier, Y. (2012). EVELIN - ein Forschungsprojekt zur systematischen Verbesserung des Lernens von Software Engineering. *Embedded Software Engineering (ESE)*, 653–658.

Akayama, S., Demuth, B., Lethbridge, T. C., Scholz, M., Stevens, P. & Stikkolorum, D. R. (2013). Tool use in software modelling education. In T. C. Lethbridge & P. Stevens (Hrsg.), *CEUR Workshop Proceedings* (Bd. 1134), Miami.

Akçayir, M., Akçayir, G., Pektaş, H. M. & Ocak, M. A. (2016). Augmented reality in science laboratories: The effects of augmented reality on university students' laboratory skills and attitudes toward science laboratories. *Computers in Human Behavior*, *57*, 334–342. doi:10.1016/j.chb.2015.12.054

van den Akker, J. (1999). Principles and Methods of Development Research. In J. van den Akker, R. Branch, K. Gustafson, N. Nieveen & T. Plomp (Hrsg.), *Design Approaches and Tools in Education and Training*. doi:10.1007/978-94-011-4255-7_1

van den Akker, J., Bannan, B. B., Kelly, A. E., Nieveen, N. & Plomp, T. (2010). An Introduction to Educational Design Research. In T. Plomp & N. Nieveen (Hrsg.), *Proceedings of the seminar conducted at the East China Normal University, Shanghai (PR China), November 23-26, 2007SLO*, Enschede: SLO Netherlands institute for curriculum development.

Ali, N. H. & Terengganu, K. (2007). A Design of an Assessment System for UML Class Diagram Zarina Shukur. *Fifth International Conference on Computational Science and Applications*, *5*, 539–544. doi:10.1109/ICCSA.2007.31

Anke, J. & Bente, S. (2019). UML in der Hochschullehre: Eine kritische Reflexion. In V. Thurner, O. Radfelder & K. Vosseberg (Hrsg.), *CEUR Workshop Proceedings* (S. 8–20). München.

Arendt, T., Mantz, F. & Taentzer, G. (2009). *UML Model Quality Assurance Techniques*. Universität Marburg. Verfügbar 29. Dezember 2020 unter

http://spes2020.informatik.tu-muenchen.de/results/AT-AP4-D-AT-4_1_g.pdf

Arnold, J., Kremer, K. & Mayer, J. (2017). Scaffolding beim Forschenden Lernen. *Zeitschrift für Didaktik der Naturwissenschaften*, *23*(1), 21–37. doi:10.1007/s40573-016-0053-0

Artino, A. R. (2005). *Review of the MSLQ*. University of Conneticut. Conneticut. Verfügbar 25. Dezember 2020 unter https://files.eric.ed.gov/fulltext/ED499083.pdf

Arumugam, R. & Shanmugamani, R. (2018). *Hands-On Natural Language* (P. Dhandre, A. Singh, S. Kolte, S. Nikalje & S. Editing, Hrsg.). Birmingham: Packt Publishing.

Atkinson, R. K., Derry, S. J., Renkl, A. & Wortham, D. (2000). Learning from examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research*, *70*(2), 181–214. doi:10.3102/00346543070002181

Ayres, P. & Paas, F. (2012). Cognitive load theory: New Directions and Challenges. *Applied Cognitive Psychology*, *26*(6), 827–832. doi:10.1002/acp.2882

Bacca, J., Baldiris, S., Fabregat, R., Graf, S. & Kinshuk. (2014). Augmented reality trends in education: A systematic review of research and applications. *Educational Technology and Society*, *17*(4), 133–149.

Or-Bach, R. & Lavy, I. (2004). Cognitive activities of abstraction in object orientation: An empirical study. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, *36*(2), 82–86. doi:10.1145/1024338.1024378

Bälter, O. (2017). Moving Technology - Enhanced - Learning Forward: Bridging Divides through Leadership. *The International Review of Research in Open and Distributed Learning*, *18*(3). Verfügbar 29. Dezember 2020 unter https://doi.org/10.19173/irrodl.v18i3.3250

Balzert, H. (2001). *UML kompakt mit Checklisten*. Heidelberg, Berlin: Spektrum Akademischer Verlag.

Balzert, H. (2011). *Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb* (3. Aufl.) (H. Balzert, Hrsg.). doi:10.1007/978-3-8274-2246-0

Balzert, H. & Balzert, H. (2009). *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering* (3. Aufl.) (H. Balzert, Hrsg.). Heidelberg: Spektrum.

Bartel, A. (2018). *Konzeption und Entwicklung eines DSM-basierten Gamification Authoring Systems zur Unterstützung Hochschulischer Lehre Inaugural-Dissertation* (Dissertation, Universität Regensburg, Regensburg). doi:10.5283/epub.38076

Bass, L., Clements, P. & Kazman, R. (2003). *Software architecture in practice*. Boston: Addison-Wesley Professional.

Baumert, J. (1993). Lernstrategien, motivationale Orientierung und Selbstwirksamkeitsüberzeugungen im Kontext schulischen Lernens. *Unterrichtswissenschaft*, *21*(4), 327–354. Verfügbar 29. Dezember 2020 unter http://www.juventa.de

Becker, H. S. & Geer, B. (1979). Teilnehmende Beobachtung: Die Analyse qualitativer Forschungsergebnisse. In C. Hopf & E. Weingarten (Hrsg.), *Qualitative Sozialforschung* (3. Aufl., S. 139–168). Stuttgart: Klett-Cotta.

Bednarik, R., Budde, L. & Vrzakova, H. (2018). Eye-movement Modeling Examples in Source Code Comprehension: A Eye-movement Modeling Examples in Source Code Comprehension: A Classroom Study. In *Proceedings of the 18th Koli Calling International Conference on Computing Education Research* (S. 1–8). doi:10.1145/3279720.3279722

Biggs, J. B., Collis, K. F. & Edward, A. J. (1982). *Evaluating the Quality of Learning: The SOLO Taxonomy (Structure of the Observed Learning Outcome)*. New York: Academic Press.

Bilandzic, H. & Trapp, B. (2000). *Qualitative Kinder- und Jugendmedienforschung. Theorie und Methoden: ein Arbeitsbuch* (B. Schorb & I. Paus-Haase, Hrsg.). München: KoPäd.

Boerner, S., Seeber, G., Keller, H. & Beinborn, P. (2005). Lernstrategien und Lernerfolg im Studium: Zur Validierung des LIST bei berufstätigen Studierenden. *Zeitschrift fur Entwicklungspsychologie und Padagogische Psychologie*, *37*(1), 17–26. doi:10.1026/0049-8637.37.1.17

Bolloju, N. & Leung, F. S. K. (2006). Assisting novice analysts in developing quality conc. *Communications of the ACM*, *49*(7), 108–112. doi:10.1145/1139922.1139926

Bologna Follow-Up Group. (2014). ECTS USERS' GUIDE 2015 Draft Version, 1–58. doi:10.2766/87592

Booch, G., Jacobson, I. & Rumbaugh, J. (2001). OMG unified modeling language specification. *Object Management Group*.

Booch, G., Rumbaugh, J. & Jacobson, I. (1999). *The Unified Modelling Language User Guide*. Massachusetts: Addison-Wesley.

Booch, G. & Eykholt, E. (1996). *Best of Booch: Designing Strategies for Object Technology* (E. M. E. Grady Booch, Hrsg.). Cambridge: Cambridge University Press.

Bortz, J. & Schuster, C. (2010). *Statistik für Human- und Sozialwissenschaftler*. Berlin, Heidelberg: Springer.

Boticki, I., Barisic, A., Martin, S. & Drljevic, N. (2012). Sortko: Learning Sorting Algorithms with Mobile Devices. In *2012 IEEE Seventh*

International Conference on Wireless, Mobile and Ubiquitous Technology in Education (S. 49–56). doi:10.1109/wmute.2012.15

Bourque, P., Dupuis, R., Abran, A., Moore, J. W. & Tripp, L. (1999). The Guide to the software engineering body of knowledge. *16*(6), 35–44. doi:10.1109/52.805471

Brabrand, C. & Dahl, B. (2009). Analyzing CS competencies using the SOLO taxonomy. *Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE*, 1. doi:10.1145/1562877.1562879

Brandes, D. & Ginnis, P. (1996). *A guide to student-centred learning*. Cheltenham: Stanley Thornes.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M. & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, *80*, 571–583. doi:10.1016/j.jss.2006.07.009

Brinkerhoff, R. O., Brethower, D. M., Nowakowski, J. & Hluchyj, T. (2012). *Program evaluation: A practitioner's guide for trainers and educators*. Boston: Kluwer-Nijhoff Publishing.

Brosius, H.-B., Haas, A. & Koschel, F. (2016). *Methoden der empirischen Kommunikationsforschung. Eine Einführung* (7. Aufl.) (G. Bentele, H.-B. Brosius & O. Jarren, Hrsg.). Wiesbaden: Springer VS.

Brünken, R., Plass, J. L. & Leutner, D. (2003). Direct measurement of cognitive load in multimedia learning. *Educational Psychologist*, *38*(1), 53–61. doi:10.1207/S15326985EP3801_7

Buber, R. (2007). *Qualitative Marktforschung. Konzepte-Methoden-Analysen* (1. Aufl.) (R. Buber & H. H. Holzmüller, Hrsg.). Wiesbaden: Gabler.

Chalk, P. (2000). Apprenticeship learning of software engineering using Webworlds. In *Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE* (S. 112–115).

Chandler, P., Kalyuga, S., Tuovinen, J. & Sweller, J. (2001). When Problem Solving Is Superior to Studying Worked Examples. *Journal of Educational Psychology*, *93*, 579–588. doi:10.1037/0022-0663.93.3.579

Chandler, P. & Sweller, J. (1991). Cognitive Load Theory and the Format of Instruction. *Cognition and Instruction*, *8*(4), 293–332. doi:10.1207/s1532690xci0804_2

Chaudron, M. R. V., Heijstek, W. & Nugroho, A. (2012). How effective is UML modeling?: An empirical perspective on costs and benefits. *Software and Systems Modeling*, *11*(4), 571–580. doi:10.1007/s10270-012-0278-4

Cheers, H., Javed, M., Lin, Y. & Smith, S. P. (2019). Exploring a Comprehensive Approach for the Automated Assessment of UML. In

8th International Congress on Advanced Applied Informatics (IIAI-AAI) (S. 133–139). doi:10.1109/IIAI-AAI.2019.00036

Chren, S., Buhnova, B., Macak, M., Daubner, L. & Rossi, B. (2019). Mistakes in UML Diagrams: Analysis of Student Projects in a Software Engineering Course. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)* (S. 100–109). doi:10.1109/ICSE-SEET.2019.00019

Cobb, P. (1994). Where Is the Mind? Constructivist and Sociocultural Perspectives on Mathematical Development. *Educational Researcher*, *23*(October), 13–20. doi:10.3102/0013189X023007013

Cockburn, A. (2003). *Use Cases effektiv erstellen* (1. Aufl.). Bonn: mitp Verlags GmbH & Co. KG.

Colburn, T. & Shute, . E. G. (2007). Abstraction in Computer Science. *Minds & Machines*, *17*, 169–184. doi:10.1007/s11023-007-9061-7

Collins, A. (1987). *Cognitive Apprenticeship: Teaching The Craft of Reading, Writing, and Mathematics*. University of Illinois At Urbana-Champaign. Champaign.

Cooper, P. A. (1993). Paradigm Shifts in Designed Instruction: From Behaviorism to Cognitivism to Constructivism. *Educational Technology*, *33*(5), 12–19. Verfügbar 29. Dezember 2020 unter https://www.learntechlib.org/p/170882

Denzin, N. K. (1989). The Research Act: A Theoretical Introduction to Sociological Methods. *17*(4), 500. doi:10.2307/1318434

Dewey, J. (1902). The Child and the Curriculum. *University of Chicago Press*.

DIN Deutsches Institut für Normung e. V. (2006). *DIN EN ISO 9241 Ergonomie der Mensch-System-Interaktion*. Berlin: Beuth Verlag.

DIN Deutsches Institut für Normung e. V. (2020). *Ergonomie der Mensch-System-Interaktion – Teil 210: Menschzentrierte Gestaltung interaktiver Systeme (DIN EN ISO 9241-210:2019)*. In DIN Deutsches Institut für Normung e. V. (Hrsg.), *DIN EN ISO 9241 Ergonomie der Mensch-System-Interaktion*. Berlin: Beuth Verlag.

Dobing, B. & Parsons, J. (2008). Dimensions of UML diagram use: A survey of practitioners. *Journal of Database Management*, *19*(1), 1–18. doi:10.4018/jdm.2008010101

Dominic, B., Julián, D., Alex, H., Hernan, d. L., Grigori, M., Fernando, S. & Mani, S. (2013). *Developer's Guide to Microsoft Enterprise Library* (2. Aufl.). Microsoft.

Dörge, C. (2015). *Informatische Schlüsselkompetenzen. Konzepte der Informationstechnologie im Sinne einer informatischen Allgemeinbildung* (Diss., Universität Potsdam, Potsdam).

Döring, N. & Bortz, J. (2016). *Forschungsmethoden und Evaluation in den Sozial- und Humanwissenschaften* (5. Aufl.). doi:10.1007/978-3-642-41089-5

Du Prel, J. B., Röhrig, B., Hommel, G. & Blettner, M. (2010). Auswahl Statistischer Testverfahren. *Serie zur Bewertung Wissenschaftlicher Publikationen*, *107*(19), 343–348. doi:10.3238/arztebl.2010.0343

Duffy, T. M. & Donald, C. (1996). Constructivism : Implications for the Design and Delivery of Instruction. *Handbook of Research for Educational Communications and Technology*, *171*(4), 170–198. arXiv: arXiv:1408.1149

Duschl, K., Obermeier, M. & Vogel-Heuser, B. (2014). An experimental study on UML Modeling errors and their causes in the education of model driven PLC programming. In *IEEE Global Engineering Education Conference, EDUCON*. doi:10.1109/EDUCON.2014.6826078

Ebert, M. W. (2018). *Webbasierte Ad-hoc-Programmieraufgaben zur Vermittlung von grundlegenden Konzepten der Programmierung in Vorlesungen* (Diss., Universität Bayreuth, Bayreuth). doi:10.15495/EPub_UBT_00004160

Eckerdal, A., McCartney, R., Moström, J. E., Ratcliffe, M. & Zander, C. (2007). Can graduating students design software systems? *Proceedings of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*, 403–407. doi:10.1145/1121341.1121468

Elliott, J. (1991). Action Research for Educational Change. Developing teachers and teaching. Philadelphia: Open University Press.

Erickson, J. & Siau, K. (2007). Can UML be simplified? Practitioner use of UML in separate domains. In *CEUR Workshop Proceedings* (Bd. 365, S. 81–90). Trondheim.

Espinosa, E. D., Noguez, J., Benes, B. & Bueno, A. (2005). POLizied e-Learning using contract management. *Computers & Education*, *45*(1), 75–103. doi:https://doi.org/10.1016/j.compedu.2004.04.012

Europäisches Parlament & Rat der Europäischen Union. (2016). Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates. Verfügbar 29. Dezember 2020 unter https://www.bmjv.de/DE/Themen/FokusThemen/DSGVO/_documents/Amtsblatt_EU_DSGVO.html#:~:text=Sie%20sind%20hier:-,Verordnung%20(EU)%202016/679%20des%20Europ%C3%A4ischen%20Parlaments%20und%20des,/EG%20(Datenschutz-Grundverordnung)

Feldgen, M. & Clua, O. (2012). Promoting design skills in distributed systems. *Proceedings - Frontiers in Education Conference, FIE*. doi:10.1109/FIE.2012.6462229

Fielding, N. & Fielding, J. (1986). *Linking Data*. doi:10.4135/9781412984775

Figl, K., Mendling, J., Strembeck, M. & Recker, J. (2010). On the cognitive effectiveness of routing symbols in process modeling languages. *Lecture Notes in Business Information Processing*, *47 LNBIP*(May), 230–241. doi:10.1007/978-3-642-12814-1_20

Flagg, B. N. (1990). *Formative evaluation for educational technologies*. New York: Routledge.

Flick, U. (1987). Methodenangemessene Gütekriterien in der qualitativ-interpretativen Forschung. In J. Bergold & U. Flick (Hrsg.), *Ein-Sichten Zugänge zur Sicht des Subjekts mittels qualitativer Forschung* (S. 247–262). Tübingen: dgvt-Verlag.

Forschergruppe Kassel. (2006). Archimedes und die Sache mit der Badewanne. Gestufte Hilfen im naturwissenschaftlichen Unterricht. *Friedrich Jahresheft*, 84–88.

Frank, U. (2000). Die Unified Modeling Language (UML) - ein bedeutsamer Standard für die konzeptionelle Modellierung. *Das Wirtschaftsstudium (wisu)*, *5*(29), 709–718.

Frank, U. & Prasse, M. (1997). *Ein Bezugsrahmen zur Beurteilung objektorientierter Modellierungssprachen - veranschaulicht am Beispiel von OML und UML* (Techn. Ber. Nr. 6). Universität Duisburg Essen. Verfügbar 25. Dezember 2020 unter https://www.wi-inf.uni-due.de/FGFrank/documents/Arbeitsberichte_Koblenz/Nr6.pdf

Franke-Braun, G. (2004). *Aufgaben mit gestuften Lernhilfen. Ein Aufgabenformat zur Förderung der sachbezogenen Kommunikation und Lernleistung für den naturwissenschaftlichen Unterricht* (Diss., Universität Kassel).

Frezza, S. & Andersen, W. (2006). Interactive exercises to support effective learning of UML structural modeling. In *Proceedings - Frontiers in Education Conference, FIE* (S. 1–6). doi:10.1109/FIE.2006.322717

Friedrich, H. F. & Mandl, H. (1992). Lern-und Denkstrategien. Ein Problemaufriss. *Lern-und Denkstrategien. Analyse und Intervention*, 3–54.

Frommann, U. (2005). Die Methode Lautes Denken. Tübingen. Verfügbar 25. Dezember 2020 unter https://www.e-teaching.org/didaktik/qualitaet/usability/Lautes%20Denken_e-teaching_org.pdf

Fuchs, F. (2020). User Centered Design. Verfügbar 13. Dezember 2020 unter https://caderadesign.de/de/leistungen/usercentereddesign

Ganegedara, T. (2018). *Natural Language Processing with TensorFlow* (F. Pohlmann, R. Atitkar, C. Nelson, B. Rai & T. Jacob, Hrsg.). Birmingham: Packt Publishing. Verfügbar 29. Dezember 2020 unter https://github.com/PacktPublishing/Natural-Language-Processing-with-TensorFlow

Garrison, D. R., Anderson, T. & Archer, W. (1999). Critical Inquiry in a Text-Based Environment: Computer Conferencing in Higher Educa-

tion. *Internet and Higher Education*, *2*(2-3), 87–105. doi:10.1016/S1096-7516(00)00016-6

Garrison, D. R., Anderson, T. & Archer, W. (2001). Critical Thinking, Cognitive Presence, and Computer Conferencing in Distance Education. *American Journal of Distance Education*, *15*, 7–23. doi:https://doi.org/10.1080/08923640109527071

Gašević, D., Adesope, O., Joksimović, S. & Kovanović, V. (2015). Externallyfacilitated regulation scaffolding and role assignment to develop cognitive presence in asynchronous online discussions. *The Internet and Higher Education*, *24*, 53–65. doi:10.1016/j.iheduc.2014.09.006

Geirhos, M. (2015). *Entwurfsmuster: das umfassende Handbuch* (1. Aufl.). Bonn: Rheinwerk-Verlag.

Gelhausen, T., Landh, M. & Sven, J. K. (2008). Automatic Checklist Generation for the Assessment of UML Models. *International Conference on Model Driven Engineering Languages and Systems*, 387–399. doi:10.1007/978-3-642-01648-6_40

von Goethe, J. W. (1810). *Die Wahlverwandtschaften*. Tübingen.

van Gog, T., Jarodzka, H., Scheiter, K., Gerjets, P. & Paas, F. (2009). Attention guidance during example study via the model's eye movements. *Computers in Human Behavior*, *25*(3), 785–791. doi:10.1016/j.chb.2009.02.007

Granda, M. F., Condori-Fernandez, N., Vos, T. E. J. & Pastor, O. (2015). What do we know about the defect types detected in conceptual models? In *Proceedings - International Conference on Research Challenges in Information Science* (S. 88–99). doi:10.1109/RCIS.2015.7128867

Greeno, J. G. (1989). Situations, mental models, and generative knowledge. *Complex information processing: The impact of Herbert A. Simon*, 285–318.

Grüner, F. (2011). *Lernstrategien und Prüfungsangst bei Studierenden der Studiengänge Humanmedizin und Lehramt* (Diss., JMU Würzburg). Verfügbar 29. Dezember 2020 unter http://opus.bibliothek.uni-wuerzburg.de/volltexte/2011/6473/pdf/FranziskaGruenerDiss.pdf

Haidry, S.-e.-Z., Falkner, K. & Szabo, C. (2017). Identifying Domain-Specific Cognitive Strategies for Software Engineering. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '17*, 206–211. doi:10.1145/3059009.3059032

Hannafin, M. J., Hiil, J. R. & Land, S. M. (1997). Student Centered Learning and Interactive Multimedia.pdf. *Contemporary Education*, *68*(2).

Harden, R. M. & Crosby, J. (2000). AMEE guide no 20: The good teacher is more than a lecturer - The twelve roles of the teacher. *Medical Teacher*, *22*(4), 334–347. doi:10.1080/014215900409429

Hartmann, W., Näf, M. & Reichert, R. (2006). *Informatikunterricht planen und durchführen* (1. Aufl.). Berlin Heidelberg: Springer.

Hauser, F., Reuter, R., Gegenfurtner, A., Gruber, H. G. & Mottok, J. (2019). Eye Movements in Software Modelling - What Do They Tell Us About Heuristics. *ICERI2019 Proceedings*, *1*(November), 6064–6070. doi:10.21125/iceri.2019.1469

Hazzan, O. & Kramer, J. (2016). Assessing abstraction skills. *Communications of the ACM*, *59*(12), 43–45. doi:10.1145/2926712

Heinze, T. & Thiemann, F. (1982). Kommunikative Validierung und das Problem der Geltungsbegründung. *Zeitschrift für Pädagogik*, *28*, 635–642.

Hesse, W., Barkow, G., von Braun, H., Kittlaus, H.-B. & Scheschonk, G. (1994). Terminologie der Softwaretechnik, Ein Begriffssystem für die Analyse und Modellierung von Anwendungssystemen, Teil 2: Tätigkeits- und ergebnisbezogene Elemente. *Informatik Spektrum*, *17*(2), 96–105.

Hesse, W. & Mayr, H. C. (2008). Modellierung in der Softwaretechnik: eine Bestandsaufnahme. *Informatik-Spektrum*, *31*(5), 377–393. doi:10.1007/s00287-008-0276-7

Hevner, A. R. (2007). A Three Cycle View of Design Science Research A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, *19*(2), 87–92.

Hevner, A. R. & Chatterjee, S. (2010). *Design research in information systems. Theory and Practice* (R. Sharda & S. Voß, Hrsg.). doi:10.1007/978-1-4419-5653-8

Hevner, A. R., Ram, S., March, S. & Park, J. (2004). Design Science in Information System Research. *MIS Quarterly: Management Information Systems*, *28*(2), 75–106.

Higgins, C., Mtenzi, F., O'Leary, C., Hanratty, O. & McAvinia, C. (2017). A Software Development Process for Freshman Undergraduate Students. In M. Tatnall Arthurand Webb (Hrsg.), *Tomorrow's Learning: Involving Everyone. Learning with and about Technologies and Computing. WCCE 2017. IFIP Advances in Information and Communication Technology* (Bd. 515, S. 599–608). doi:10.1007/978-3-319-74310-3_60

Hirsch, E. D. (1968). *Validity in Interpretation*. Yale: Yale University Press. Verfügbar 29. Dezember 2020 unter http://www.jstor.org/stable/j.ctt32bd9k

Hislop, G. W. & Ellis, H. J. C. (2009). Using scaffolding to improve written communication of software engineering students. *ITNG 2009 - 6th International Conference on Information Technology: New Generations*, 707–712. doi:10.1109/ITNG.2009.31

Hitz, M. & Kappel, G. (2003). *UML@Work: objektorientierte Modellierung mit UML 2*. Heidelberg: Dpunkt. verlag.

Hitz, M., Kappel, G., Kapsammer, E. & Retschitzegger, W. (2005). *UML@ Work. Objektorientierte Modellierung mit der UML 2* (3. Aufl.). Heidelberg: dpunkt.Verlag.

Hochschulrektorenkonferenz. (2015). *Standards und Leitlinien für die Qualitätssicherung im Europäischen Hochschulraum (ESG): Standards and guidelines for quality assurance in the European Higher Education Area (ESG)* (Hochschulrektorenkonferenz, Hrsg.). Bonn.

Holland, S., Griffiths, R. & Woodman, M. (1997). Avoiding object misconceptions. *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)*, *29*(1), 131–134. doi:10.1145/268085.268132

Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H. & van de Weijer, J. (2013). *Eye Tracking A Comprehensive Guide to Methods and Measures*. Oxford: Oxford University Press. arXiv: arXiv:1011.1669v3. Verfügbar 29. Dezember 2020 unter http://ukcatalogue.oup.com/product/9780199697083.do

Hussein, H. E. M. A., Bakar, M. H. S. A. & Jones, B. (2015). A Cross-Cultural Validation of the MUSIC Model of Academic Motivation and Its Associated Inventory Among Egyptian University Students. doi:10.13140/RG.2.1.2532.9123

Hutzler, I. (2018). *Wird die Substantiv-Verb-Analyse zur Generierung von Klassendiagrammen verwendet? – Eine Eye-Tracking-Studie* (Masterarbeit, OTH Regensburg, Regensburg).

Hutzler, I., Hauser, F., Reuter, R., Mottok, J. & Gruber, H. (2018). Will the Noun/Verb Analysis Be Used To Generate Class Diagrams? an Eye Tracking Study. *ICERI2018 Proceedings*, *1*, 505–514. doi:10.21125/iceri.2018.1103

Jack, T. D. (1979). Mixing qualitative and quantitative methods: Triangulation in action. *Administrative Science Quarterly*, *24*(4), 602–611. Verfügbar 29. Dezember 2020 unter https://www.jstor.org/stable/2392366?seq=1

Jarodzka, H., Balslev, T., Holmqvist, K. & Nystro, M. (2012). Conveying clinical reasoning based on visual observation via eye-movement modelling examples. *Instructional Science*, *40*, 813–827. doi:10.1007/s11251-012-9218-5

Jarodzka, H., Van Gog, T., Dorr, M., Scheiter, K. & Gerjets, P. (2013). Learning to see: Guiding students' attention via a Model's eye movements fosters learning. *Learning and Instruction*, *25*, 62–70. doi:10.1016/j.learninstruc.2012.11.004

Jonassen, D. (1999). Designing constructivist learning environments. *Instructional-Design Theories and Models: A New Paradigm of Instructional Theory*, *2*, 215–239.

Jonassen, D. H. (1985). Objectivism versus Constructivism : Do We Need a New Philosophical Paradigm ? *ETR&D*, *39*, 5–14.

Jonassen, D. H. (1997). Instructional Design Models for Well-Structured and Ill-Structured Problem-Solving Learning Outcomes. *ETR&D*, *45*(1), 65–94. Verfügbar 29. Dezember 2020 unter http://www1.folha.uol.com.br/ciencia/880408-bahia-inicia-uso-de-inseto-transgenico-contra-dengue.shtml

Jones, B. D. (2009). Motivating Students to Engage in Learning : The MUSIC Model of Academic Motivation. *International Journal of Teaching and Learning in Higher Education*, *21*(2), 272–285.

Jones, B. D., Li, M. & Cruz, J. M. (2017). A Cross-Cultural Validation of the MUSIC® Model of Academic Motivation Inventory: Evidence from Chinese- and Spanish-Speaking University Students. *International Journal of Educational Psychology*, *6*(1), 25. doi:10.17583/ijep.2017.2357

Jones, B. D. & Skaggs, G. (2016). Measuring Students' Motivation: Validity Evidence for the MUSIC Model of Academic Motivation Inventory. *International Journal for the Scholarship of Teaching and Learning*, *10*(1). doi:10.20429/ijsotl.2016.100107

Jurafsky, D. & Martin, J. H. (2019, 16. Oktober). *Speech and Language Processing. An Introduction to Natural Language Processing,Computational Linguistics, and Speech Recognition*. Stanford University und University of Colorado at Boulder. Stanford. Verfügbar 25. Dezember 2020 unter https://web.stanford.edu/~jurafsky/slp3/edbook_oct162019.pdf

Kehrwald, B. A. & McCallum, F. (2015). Degrees of change: Understanding academics experiences with a shift to flexible technology-enhanced learning in initial teacher education. *Australian Journal of Teacher Education*, *40*(7), 43–56. doi:10.14221/ajte.2015v40n7.4

Kember, D. (1997). A reconceptualisation of the research into university academics' conceptions of teaching. *Learning and Instruction*, *7*(3), 255–275. doi:10.1016/S0959-4752(96)00028-X

Kips, D. (2013). Analyse und Design objektorientierter Softwaresysteme mit der Unified Modeling Language Teil 2 [Vorlesung]. FAU Erlangen-Nürnberg, Erlangen-Nürnberg.

Kirk, J. & Miller, M. (1986). Reliability and Validity in Qualitative Research. doi:10.4135/9781412985659

Kirkwood, A. & Price, L. (2014). Technology-enhanced learning and teaching in higher education: what is 'enhanced' and how do we

know? A critical literature review. *Learning, Media and Technology*, *39*(1), 6–36. doi:10.1080/17439884.2013.770404

Kitchenham, B. & Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information and Software Technology*, *55*(12), 2049–2075. doi:10.1016/j.infsof.2013.07.010

Kitchenham, B. & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

Klüver, J. (1997). *Kommunikative Validierung – einige vorbereitende Bemerkungen zum Projekt ‚Lebensweltanalyse von Fernstudenten'*. Werkstattbericht FernUniversität Hagen.

Knorr, P. & Schramm, K. (2012). Datenerhebung durch Lautes Denken und Lautes Erinnern in der fremdsprachendidaktischen Empirie. Grundlagenbeitrag. In S. Doff (Hrsg.), *Fremdsprachenunterricht empirisch erforschen. Grundlagen - Methoden - Anwendung.* (S. 184–201). Tübingen: Narr Francke Attempto.

Konrad, K. (2010). Lautes Denken. In G. Mey & K. Mruck (Hrsg.), *Handbuch qualitative Forschung in der Psychologie* (1. Aufl., S. 476–490). Wiesbaden: VS Verlag für Sozialwissenschaften.

Kramer, J. (2006). Abstraction in Computer Science & Software Engineering: A Pedagogical Perspective. *System Design Frontier Journal*, *3*(12), 1–9.

Kramer, J. (2007). Is Abstraction the key to computing? Abstraction: What is it? Why is it so important? *Communications of the ACM*, *50*(4), 37–42.

Krippendorff, K. (1980). *Content analysis: an introduction to its methodology*. Beverly Hills: Sage Publications.

Kruchten, P. (1995). Architectural Blueprints—The "4+ 1" View Model of Software Architecture. *Tutorial Proceedings, Tri-Ada'95*, *12*, 540–555.

Kruchten, P. (2004). *The Rational Unified Process. An Introduction* (3. Aufl.). Boston: Addison-Wesley Professional.

Kruus, H., Robal, T. & Jervan, G. (2014). Teaching modeling in SysML/UML and problems encountered. In *Proceedings of the 25th International Conference on European Association for Education in Electrical and Information Engineering, EAEEIE 2014*. doi:10.1109/EAEEIE.2014.6879380

Kühnel, A. (2013). *Visual C# 2012: Das umfassende Handbuch* (6. Aufl.). Bonn: Rheinwerk Computing.

Kvale, S. (1987). Validity in the qualitative research interview. Interviewet som forskningsmetode. *Psykologisk Skriftserie*, *12/1*(1), 68–104.

Langer, P., Mayerhofer, T., Wimmer, M. & Kappel, G. (2014). On the usage of UML: Initial results of analyzing open UML models. *Lecture*

Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI), 289–304.

Lea, S. J., Stephenson, D. & Troy, J. (2003). Higher education students' attitudes to student-centred learning: Beyond 'educational bulimia'? *Studies in Higher Education*, *28*(3), 321–334. doi:10.1080/03075070309293

Leinonen, T., Toikkanen, T. & Silfvast, K. (2008). Software as hypothesis: research-based design methodology. In *Proceedings of the Tenth Anniversary Conference on Participatory Design 2008* (S. 61–70). Bloomington, Indiana: Indiana University.

Lincoln, T. S. & Guba, E. G. (1979). The Distinction Btween Merit and Worth in !Valuation. In *5th Annual Meeting of the Evaluation Network* (S. 24–26). Cincinnati, Ohio.

Lindland, I., Guttorm, S. & Solvberg, A. (1994). Understanding Quality in Conceptual Modelling. *IEEE Software*, *1*, 42–49. doi:10.1109/52.268955

Linn, M. C. (1992). How Can Hypermedia Tools Help Teach Programming? *International Journal of Man-Machine Studies*, *2*(4), 119–139. doi:10.1016/0959-4752(92)90027-J

Linn, M. C. & Clancy, M. (1992). Can Experts' Explanations Help Students Develop Program Design Skills? *Learning and Instruction*, *36*(2), 511–550. doi:10.1016/0959-4752(92)90027-J

Litchfield, D., Ball, L., Donovan, T., Manning, D. J. & Crawford, T. (2010). Viewing another person' s eye movements improves identification of pulmonary nodules in chest xray inspection. *Journal of experimental psychology*, *16*(3), 251–262. doi:10.1037/a0020082

Lompscher, J. (2005). Erfassung von Lernstrategien auf der Reflexionsebene. *LLF-Berichte / Universität Potsdam, Zentrum für Lehrerbildung*, *13*. Verfügbar 29. Dezember 2020 unter https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/417/file/LOMERFAS.pdf

Ludi, S., Natarajan, S. & Reichlmayr, T. (2005). An introductory software engineering course that facilitates active learning. In *Proceedings of the Thirty-Sixth SIGCSE Technical Symposium on Computer Science Education, SIGCSE 2005* (S. 302–306). Association for Computing Machinery.

Manno, I., Palmieri, G. & Scarano, V. (2011). Collaborative diagram drawing: A case study on scaffolding self-regulated behaviors. In *CEUR Workshop Proceedings* (Bd. 777, S. 19–24).

Manouselis, N., Drachsler, H., Vuorikari, R., Hummel, H. & Koper, R. (2011). Recommender Systems in Technology Enhanced Learning.

Recommender Systems Handbook, 387–415. doi:10.1007/978-1-4939-0530-0

Martsch, M. & Schulz, A. (2015). Entwicklung von Lernstrategien durch Blended Learning in der betrieblichen Ausbildung Berufliche Lehr-Lernforschung. *Berufs- und Wirtschaftspädagogik*, *28*. Verfügbar 24. Dezember 2020 unter https://www.bwpat.de/ausgabe/28/martsch-schulz

Mason, L., Pluchino, P. & Tornatora, M. C. (2015). Eye-movement modeling of integrative reading of an illustrated text: Effects on processing and learning. *Contemporary Educational Psychology*, *41*, 172–187. doi:10.1016/j.cedpsych.2015.01.004

Mason, L., Pluchino, P. & Tornatora, M. C. (2016). Using eye-tracking technology as an indirect instruction tool to improve text and picture processing and learning. *British Journal of Educational Technology*, *47*(6), 1083–1095. doi:10.1111/bjet.12271

Matter, B. (2017). *Lernen in heterogenen Lerngruppen Erprobung und Evaluation eines Konzepts für den jahrgangsgemischten* (B. Barzel, A. Büchter, B. Rott, F. Schacht & P. Scherer, Hrsg.). Wiesbaden: Springer Spektrum.

Matthews, R., Hin, H. S. & Choo, K. A. (2015). Practical Use of Review Question and Content Object as Advanced Organizer for Computer Programming Lessons. *Procedia - Social and Behavioral Sciences*, *172*, 215–222. doi:10.1016/j.sbspro.2015.01.357

Mayer, R. E. (1984). Aids to text comprehension. *Educational Psychologist*, *19*(1), 30–42. doi:10.1080/00461528409529279

Mayer, R. E. (1989). Systematic Thinking Fostered by Illustrations in Scientific Text. *Journal of Educational Psychology*, *81*(2), 240–246. doi:10.1037/0022-0663.81.2.240

Mayer, R. E. (1997). Multimedia learning: Are we asking the right questions? *Educational Psychologist*, *32*(1), 1–19. doi:10.1207/s15326985ep3201_1

Mayer, R. E. (2014). Cognitive theory of multimedia learning. *The Cambridge Handbook of Multimedia Learning, Second Edition*, 43–71. doi:10.1017/CBO9781139547369.005

Mayer, R. E. & Anderson, R. B. (1991). Animations need narrations: An experimental test of a dual-coding hypothesis. *Journal of Educational Psychology*, *83*(4), 484–490. doi:10.1037/0022-0663.83.4.484

Mayer, R. E. & Gallini, J. K. (1990). When Is an Illustration Worth Ten Thousand Words? *Journal of Educational Psychology*, *82*(4), 715–726. doi:10.1037/0022-0663.82.4.715

Mayer, R. E. & Moreno, R. (1998). A split-attention effect in multimedia learning: Evidence for dual processing systems in working memory.

Journal of Educational Psychology, *90*(2), 312–320. doi:10.1037/0022-0663.90.2.312

Mayring, P. (1991). Qualitative Inhaltsanalyse. In E. v. K. U. Flick, H. Keupp, L. v. Rosenstiel & S. Wolff (Hrsg.), *Handbuch qualitative Forschung : Grundlagen, Konzepte, Methoden und Anwendungen* (S. 209–213). München: Beltz.

Mayring, P. (2016). *Einführung in die Qualitative Sozialforschung* (6. Aufl.). Weinheim und Basel: Beltz.

Mc Kenney, S. & Reeves, T. C. (2010, 7. November). *Conducting Educational Design Research* (2. Aufl.). London: Routledge Taylor & Francis Group.

van Merriënboer, J. J. G., Kirschner, P. A. & Kester, L. (2003). Taking the Load off a Learner's Mind: Instructional Design for Complex Learning. *Educational Psychologist*, *38*(1), 5–13.

Metcalf, S. J., Chen, J. A., Kamarainen, A. M., Frumin, K. M., Vickrey, T. L., Grotzer, T. A. & Dede, C. J. (2019). Transitions in Student Motivation During a MUVE-Based Ecosystem Science Curriculum: An Evaluation of the Novelty Effect. In K. Becnel (Hrsg.), *Emerging Technologies in Virtual Learning Environments* (S. 96–115). doi:10.4018/978-1-5225-7987-8.ch005

Mohammed, S. & Abdelazziz, A. M. (2013). WIDE an interactive web integrated development environment to practice C programming in distance education. *Proceedings - 2013 1st International Conference of the Portuguese Society for Engineering Education, CISPEE 2013*. doi:10.1109/CISPEE.2013.6701964

Müller, L. (2019). *Welche Schwierigkeiten haben Studierende mit der Modellierung von Software mithilfe von UML? – Eine Empirische Studie aus Studierendenperspektive* (Masterarbeit, OTH Regensburg, Regensburg).

Murray, S., Ryan, J. & Pahl, C. (2003). A tool-mediated cognitive apprenticeship approach for a computer engineering course. *Advanced Learning Technologies, 2003. Proceedings. The 3rd IEEE International Conference on*, 2–6. doi:10.1109/ICALT.2003.1215014

Neill, G. O. & Mcmahon, T. (2005). Student-centered Learning: What does this mean for students and lecturers? *Emerging issues in the practice of University Learning and Teaching*.

Nielsen, J. & Landauer, J. (1993). A mathematical model of finding the usability problem. Proceedings of the CHI 93 proceedings of the Interact conference on human factors in computing systems. In *CHI '93: Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems* (S. 206–213). doi:10.1145/169059.169166

Nieveen, N. & Flomer, E. (2013). Formative Evaluation in Educational Design Research. In T. Plomp & N. Nieveen (Hrsg.), *Educational Design Research* (S. 152–169). Enschede: SLO Netherlands Institute for Curriculum Development. Verfügbar 25. Dezember 2020 unter http://international.slo.nl/publications/edr/

Object Management Group, Inc. (OMG). (2017). *About the Unified Modeling Language Specification Version 2.5.1*. Version 2.5.1. Object Management Group, OMG Unified Modeling Language Publication. Verfügbar 25. Dezember 2020 unter https://www.omg.org/spec/UML/2.5.1/PDF

Odisho, O., Aziz, M. & Giacaman, N. (2016). Teaching and learning data structure concepts via Visual Kinesthetic Pseudocode with the aid of a constructively aligned app. *Computer Applications in Engineering Education*. doi:10.1002/cae.21768

Oelerich, A. (2005). *Robuste Ratingverfahren zur Steigerung der Prognosequalität quantitativer Ratingverfahren* (Diss., Universität Bremen, Wiesbaden). doi:10.1007/978-3-322-81932-1

Paas, F., Tuovinen, J. E., Tabbers, H. & Van Gerven, P. W. M. (2003). Cognitive load measurement as a means to advance cognitive load theory. *Educational Psychologist*, *38*(1), 63–71. doi:10.1207/S15326985EP3801_8

Paivio, A. (1990). *Mental representations: A dual coding approach*. Oxford: Oxford Science Publications.

Papadopoulos, P. M., Demetriadis, S. N. & Stamelos, I. G. (2007). Case-based instruction on the web for teaching software project management. *Association for Computing Machinery*, *39*(3), 136–140. doi:10.1145/1269900.1268826

Passey, D. (2019). Technology-enhanced learning: Rethinking the term, the concept and its theoretical background. *British Journal of Educational Technology*, *50*(3), 972–986. doi:10.1111/bjet.12783

Peffers, K., Tuunanen, T., Rothenberger, M. A. & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, *24*(3), 45–77. doi:10.2753/MIS0742-1222240302

Pennington, N., Lee, A. Y. & Rehder, B. (1995). Cognitive Activities and Levels of Abstraction in Procedural and Object-Oriented Design Cognitive Activities and Levels of Abstraction in Procedural and Object- Oriented Design. *Human – Computer Interaction*, *0024*, 171–226.

Piaget, J. (1950). *The Psychology of Intelligence* (2. Aufl.). London: Routledge.

Pintrich, P. R., Smith, D. A., Garcia, T. & Mckeachie, W. J. (1991). Motivated Strategies for Learning Questionnaire (MSLQ). *Mediterranean Journal of Social Sciences*, *6*(1), 156–164. doi:10.5901/mjss.2015.v6n1p156

Pintrich, P. R., Smith, D. A. F., Garcia, T. & Mckeachie, W. J. (1993). Reliability and Predictive Validity of the Motivated Strategies for Learning Questionnaire (Mslq). *Educational and Psychological Measurement*, *53*(3), 801–813. doi:10.1177/0013164493053003024

Polya, G. (1985). *How to Solve It: A New Aspect of Mathematical Method*. (2. Aufl.). Princeton: Princeton University Press.

Polya, G. (2010). *Schule des Denkens. Vom Lösen Mathematischer Probleme* (4. Aufl.). Tübingen: Francke Verlag.

Prechelt, L. (2005). *Softwaretechnik : Einige Grundlagen*. Institut für Informatik, Freie Universität Berlin. Berlin.

Rational Software Corporation. (n. d.). Template for the RUP Artifact: Vision. Verfügbar 15. Oktober 2020 unter https://sceweb.uhcl.edu/helm/RUP_school_example/wcsoftwareprocessweb/templates/requirem/pt_vision.htm

Reason, J. T. (1991). *Human error*. Cambridge [England]; New York: Cambridge University Press.

Reddy, P. D., Iyer, S. & Sasikumar, M. (2017). FATHOM: TEL Environment to Develop Divergent and Convergent Thinking Skills in Software Design. *Proceedings - IEEE 17th International Conference on Advanced Learning Technologies, ICALT 2017*, 414–418. doi:10.1109/ICALT.2017.83

Reeves, W. (1999). Learner-Centered Design: A Cognitive View of Managing Complexity in Product, Information, and Environmental Design. doi:10.4135/9781452233710

Reggio, G., Leotta, M., Ricca, F. & Clerissi, D. (2013). What are the used UML diagrams? A preliminary survey. *CEUR Workshop Proceedings*, *1078*, 3–12.

Reischmann, T. & Kuchen, H. (2019). A Web-Based E-Assessment Tool for Design Patterns in UML Class Diagrams. In *SAC '19: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (S. 2435–2444). doi:10.1145/3297280.3297520

Renkl, A. (2005). The Worked-Out Examples Principle in Multimedia Learning. In R. Mayer (Hrsg.), *The Cambridge Handbook of Multimedia Learning* (S. 229–246). doi:10.1017/cbo9780511816819.016

Resnick, L. B. (1988). Treating Mathematics as an Ill-Structured Discipline. *Research agenda for mathematics education: Teaching and assessment of mathematical problem solving*.

Reuter, R., Stark, T., Sedelmaier, Y., Landes, D., Mottok, J. & Wolff, C. (2020). Insights in Students' Problems during UML Modeling. *IEEE Global Engineering Education Conference, EDUCON*, 592–600.

Reuter, R., Hauser, F., Gold-Veerkamp, C., Mottok, J. & Abke, J. (2017). Towards a Definition and Identification of Learning Obstacles in Higher Software Engineering Education. *EDULEARN17 Proceedings*, *1*, 10259–10267. doi:10.21125/edulearn.2017.0943

Reuter, R., Hauser, F., Gold-Veerkamp, C., Stark, T., Kis, J., Mottok, J., ... Meyer, D. (2018). Towards the construction of a questionnaire for the identification of learning obstacles. (S. 457–466). doi:10.1109/EDUCON.2018.8363266

Reuter, R., Hauser, F., Muckelbauer, D., Stark, T., Antoni, E., Mottok, J. & Wolff, C. (2019). Using Augmented Reality in Software Engineering Education? First insights to a comparative study of 2D and AR UML modeling. *Proceedings of the 52nd Hawaii International Conference on System Sciences CC BY-NC-ND 4.0*, *6*, 7798–7807. doi:10.24251/hicss.2019.938

Reuter, R., Knietzsch, M., Hauser, F. & Mottok, J. (2019). Supporting abstraction skills using augmented reality? (S. 320). doi:10.1145/3304221.3325562

Rogers, C. (1983). *Freedom to Learn for the 80's*. Columbus: Merrill.

Rosson, M. B. & Carroll, J. M. (1996). Scaffolded examples for learning object-oriented design. *Communications of the ACM*, *39*(4), 46–47. doi:10.1145/227210.227223

Roy, G. G. (2006). Designing and explaining programs with a literate pseudocode. *ACM Journal on Educational Resources in Computing*, *6*(1), 1–18. doi:10.1145/1217862.1217863

Rumbaugh, J., Eddy, F., Blaha, M. & Premerlani, W. (1991). *Object-Oriented Modeling and Design*. Englewood Cliffs, N.J.: Prentice Hall International.

Rupp, C., Queins, S. & die SOPHISTen. (2012). *UML 2 glasklar* (4. Aufl.). doi:10.3139/9783446431973

Rupp, C. & die SOPHISTen. (2014). *Requirements-Engineering und -Management. Aus der Praxis von klassisch bis agil* (6. Aufl.). Nürnberg: Hanser.

Sandmann, A. (2014). Lautes Denken – die Analyse von Denk-, Lern- und Problemlöseprozessen. In D. Krüger, I. Parchmann & H. Schecker (Hrsg.), *Methoden in der naturwissenschaftsdidaktischen Forschung* (Kap. 15, S. 179–188). doi:10.1007/978-3-642-37827-0_15

Sasaki, T. (n. d.). Concurrent think-aloud protocol as a socially situated construct. *International Review of Applied Linguistics in Language Teaching*, *46*, 349–374. doi:10.1515/IRAL.2008.015

Saye, J. W. & Brush, T. (2002). Scaffolding critical reasoning about history and social issues in multimedia-supported learning environments. *Educational Technology Research and Development*, *50*(3), 77–96. doi:10.1007/BF02505026

Scheele, B. & Groeben, N. (1988). *Dialog-Konsens-Methodeen zur Rekunstruktion subjektiver Theorien. Die Heidelberger Struktur-Lege-Technik, konsensuale Ziel-Mittel-Argumentation und kommunikative Flußdiagramm-Beschreibung von Handlungen*. Tübingen: Francke.

Schenk, K. D., Vitalari, N. P. & Davis, K. S. (1998). Differences between Novice and Expert Systems Analysts: What Do We Know and What Do We Do? *Journal of Management Information Systems*, *15*(1), 9–50. Verfügbar 29. Dezember 2020 unter http://www.jstor.org/stable/40398371

Schmeck, R. R., Geisler-Brenstein, E. & Cercy, S. P. (1991). Self-Concept and Learning: The revised inventory of learning processes. *Educational Psychology*, *11*(3-4), 343–362. doi:10.1080/0144341910110310

Schmedding, D. & Vasileva, A. (2017). Reviews - ein Instrument zur Qualitätsverbesserung von UML-Diagrammen. In B. Bruegge & S. Krusche (Hrsg.), *Software Engineering Unterricht an Hochschulen (SEUH)*, Hannover.

Schmidt-Weigand, F., Franke-Braun, G. & Hänze, M. (2008). Erhöhen gestufte Lernhilfen die Effektivität von Lösungsspielen? *Unterrichtswissenschaft*, *36*(4), 365–384.

Schreistetter, S. (2021). *Learning from Gaze: Evaluating Eye-Movement Modeling Examples in Software Engineering Education* (Masterarbeit, OTH Regensburg, Regensburg).

Schwab, G. (2006). *Die schönsten Sagen des klassischen Altertums*. Bindlach: Gondrom Verlag.

Scriven, M. (1972). Die Methodologie der Evaluation. In C. Wulf (Hrsg.), *Evaluation Beschreibung und Bewertung von Unterricht, Curricula und Schulversuchen* (S. 60–91). München: R. Piper & Co. Verlag.

Sedelmaier, Y., Claren, S. & Landes, D. (2013). Welche Kompetenzen benötigt ein Software Ingenieur? *CEUR Workshop Proceedings*, *956*, 117–128.

Seidl, M., Scholz, M., Huemer, C. & Kappel, G. (2015). *UML@Classroom: An introduction to object-oriented modeling*. doi:10.1007/978-3-319-12742-2

Shabo, A., Guzdial, M. & Stasko, J. (1996). Computer science apprenticeship. (S. 77–82). Evanston, Illinois: International Society of the Learning Sciences.

Shuell, T. J. (1986). Cognitive Conceptions of Learning. *Review of Educational Research*, *56*(4), 411–436. doi:10.3102/00346543056004411

Siau, K. & Loo, P.-P. (2006). Identifying difficulties in learning UML. *Information Systems Management*, *43*, 43–51. doi:10.1201/1078.10580530/46108.23.3.20060601/93706.5

Siegel, S. (1956). *Nonparametric Statistics for the behavioral sciences* (C. T. Morgan, Hrsg.). New York: McGraw-Hill Book Company.

Sien, V. Y. (2011). An investigation of difficulties experienced by students developing unified modelling language (UML) class and sequence diagrams. *Computer Science Education*, *21*(4), 317–342. doi:10.1080/08993408.2011.630127

Sien, V. Y., Editors, G., Clarke, P. J., Seidl, M., Editors, M., Margaria, T., … Taentzer, G. (2010). Teaching Object-Oriented Modelling using Concept Maps. *Electronic Communications of the EASST*, *34*.

Silver, M. S., Markus, M. L. & Beath, C. M. (1995). The information technology interaction model: A foundation for the MBA core course. *MIS Quarterly: Management Information Systems*, *19*(3), 361–387. doi:10.2307/249600

Simon, H. A. (1996). *The Sciences of the Artificial Third edition* (3. Aufl.). London: MIT Press.

Sneed, S. H. & Fleischmann, A. (n. d.). Subjekt - Prädikat - Objekt.Ein Ansatz zur Beschreibung von informationstechnisch Gestützten Geschäftsprozessen angelehnt an die natürliche Sprache. *jcom1*, 1–33.

Sommerville, I. (2013). *Software Engineering* (9. Aufl.). München: Pearson.

Standards and Guidelines for Quality Assurance in the European Higher Education Area (ESG). (2015). Brüssel: EURASHE. Verfügbar 29. Dezember 2020 unter https://www.enqa.eu/wp-content/uploads/2015/11/ESG_2015.pdf

Stark, T. (2021). *Using EMMEs in Software Engineering Education* (Masterarbeit, Universität Regensburg).

Stikkolorum, D. R., Gomes De Oliveira Neto, F. & Chaudron, M. R. V. (2018). Evaluating didactic approaches used by teaching assistants for software analysis and design using UML. *ACM International Conference Proceeding Series*, 122–131. doi:10.1145/3209087.3209107

Stiller, K. (2007). The modality principle in multimedia learning: An Open Question: When Speech Fails to Foster Learning? In A. Osswald, M. Stempfhuber & C. Wolff (Hrsg.), *Open Innovation. Proc. 10. International Symposium for Information Science* (S. 129–144). doi:10.1017/CBO9781139547369.012

Sutch, L. (2007). "You know more than you think you do Helping participants transfer knowledge. In *Proceedings ACM SIGUCCS User Services Conference* (S. 332–334). doi:10.1145/1294046.1294124

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, *12*(2), 257–285. doi:10.1207/s15516709cog1202_4

Sweller, J. (1989). Cognitive Technology: Some Procedures for Facilitating Learning and Problem Solving in Mathematics and Science. *Journal of Educational Psychology*, *81*(4), 457–466. doi:10.1037/0022-0663.81.4.457

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. *Learning and Instruction*, *4*(4), 295–312. doi:10.1016/0959-4752(94)90003-5

Sweller, J. (2003). Human Cognitive Architecture. *Academic Press*.

Sweller, J., van Merriënboer, J. J. G. & Paas, F. (2019). Cognitive Architecture and Instructional Design: 20 Years Later. *Educational Psychology Review*, *31*(2), 261–292. doi:10.1007/s10648-019-09465-5

Sweller, J., Van Merrienboer, J. J. G. & Paas, F. G. W. C. (1998). Cognitive Architecture and Instructional Design. *Educational Psychology Review*, *10*(3), 251–296. doi:10.1023/A:1022193728205

Terhart, E. (1981). Intuition, Interpretation, Argumentation. Zum Problem der Geltungsbegründung von Interpretationen. (Intuition, interprétation, argumentation. Le problème de la justification de la validité des interprétations). *Zeitschrift für Pädagogik Weinheim*, *27*(4), 769–793.

Tessmer, M. (1993). *Planning and Conducting Formative Evaluations: Improving the Quality of Education and Training*. Abingdon: Kogan Page.

The Joint Committee on Standards of Educational Evaluation. (1994). *The Program Evaluation Standards: How to Assess Evaluations of Educational Programs* (2. Aufl.) (J. R. Sanders, Hrsg.). Thousand Oaks, California: SAGE Publications.

The Joint Task Force on Computing Curricula. (2001). Computing curricula 2001 Computer Science. *Journal on Educational Resources in Computing*, *1*(3es). doi:10.1145/384274.384275

The Joint Task Force on Computing Curricula. (2004). *Computing Curriculum - Software Engineering*. ACM, IEEE.

Thomasson, B. J., Ratcliffe, M. B. & Thomas, L. A. (2006a). Improving the tutoring of software design using case-based reasoning. *Advanced Engineering Informatics*, *20*(4), 351–362. doi:10.1016/j.aei.2006.07.002

Thomasson, B., Ratcliffe, M. & Thomas, L. (2006b). Identifying novice difficulties in object oriented design. *Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education - ITICSE '06*, 28. doi:10.1145/1140124.1140135

Tobii. (2018). *Eye tracker data quality report: Accuracy, precision and detected gaze under optimal conditions - controlled environment*.

Trochim, W. M. K. (1989). An introduction to concept mapping for planning and evaluation. *Evaluation and Program Planning*, *12*(1), 1–16. doi:10.1016/0149-7189(89)90016-5

Unhelkar, B. (2005). *Verification and Validation for Quality of UML 2.0 Models*. New Jersey: Wiley-Interscience.

Urban, G., Neumann, J., Löffelmann, K. & Köller, A. (2011). *Microsoft SQL Server 2008 R2 - Das Entwicklerbuch: Grundlagen, Techniken, Profi-Know-how*. Köln: O'Reilly Verlag GmbH & Co. KG.

VanderMeer, D. & Dutta, K. (2009). Applying Learner-Centered Design Principles to UML Sequence Diagrams. *Journal of Database Management*, *20*(1), 25–47. doi:10.4018/jdm.2009010102

de Villiers, M. R. (& Harpur, P. A. (2013). Design-based research - the educational technology variant of design research: : Illustrated by the design of an m-learning environment. In *SAICSIT'13* (S. 252–261). doi:10.1145/2513456.2513471

Völzke, K. (2012). *Lautes Denken bei kompetenzorientierten Diagnoseaufgaben zur naturwissenschaftlichen Erkenntnisgewinnung*. Kassel: Kassel University Press GmbH. Verfügbar 29. Dezember 2020 unter http://nbn-resolving.de/urn:nbn:de:hebis:34-2012061541272

Vygotsky, L. S. (1979). *Mind in Society: The Development of Higher Psychological Processes*. doi:10.1525/aa.1979.81.4.02a00580

Wang, Q., Yang, S., Liu, M., Cao, Z. & Ma, Q. (2014). An eye-tracking study of website complexity from cognitive load perspective. *Decision Support Systems*, *62*, 1–10. doi:10.1016/j.dss.2014.02.007

Weinstein, C. E. & Mayer, R. E. (1983). The Teaching of Learning Strategies. *Innovation Abstracts*, *5*(32), 2–4. doi:10.1108/13555540210420989

Weinstein, C. E., Palmer, D. & Schule, A. C. (1987). *Learning and Study Strategies Inventory (LASSI)'*. Clearwater, FL: H & H Publishing.

Weskamp, S. (2019). *Heterogene Lerngruppen im Mathematikunterricht der Grundschule. Design Research im Rahmen substanzieller Lernumgebungen* (B. Barzel, A. Büchter, F. Schacht & P. Scherer, Hrsg.). Wiesbaden: Springer Spektrum. Verfügbar 25. Dezember 2020 unter http://link.springer.com/10.1007/978-3-658-25233-5

White, S., Hooper, C., Carr, L., Griffith, T., Davis, H. & Wills, G. (2006). ANNANN - Next steps for scaffolding learning about programs. *Proceedings - Sixth International Conference on Advanced Learning Technologies, ICALT 2006*, 227–231. doi:10.1109/icalt.2006.1652412

Wilcox, R. R. (2012). *Introduction to robust estimation and hypothesis testing* (3. Aufl.). Amsterdam: Academic press.

Wild, K.-P. (2005). Individuelle Lernstrategien von Studierenden. Konsequenzen für die Hochschuldidaktik und die Hochschullehre. *Beiträge zur Lehrerinnen- und Lehrerbildung*, *23*(2), 191–206.

Wild, K.-P. & Schiefele, U. (1994). Lernstrategien im Studium: Ergebnisse zur Faktorenstruktur und Reliablit{ä}t eines neuen Fragebogens. *Zeitschrift f{ü}r Differentielle und Diagnostische Psychologie*, *15*(4), 185–200.

Wittrock, M. C. (1989). Generative Processes of Comprehension. *Educational Psychologist*, *24*(4), 345–376. doi:10.1207/s15326985ep2404_2

Wuttke, E. (2000). Lernstrategien im Lernprozess. *Zeitschrift für Erziehungswissenschaft*, *3*(1), 97–110. doi:10.1007/s11618-000-0007-6

Zukunft, O. (2016). *Empfehlungen für Bachelor-und Masterprogramme im Studienfach Informatik an Hochschulen Empfehlungen für Bachelor-und Master-Programme im Studienfach Informatik an Hochschulen Inhalt*. Gesellschaft für Informatik. Verfügbar 29. Dezember 2020 unter https://dl.gi.de/handle/20.500.12116/2351

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

https://bitbucket.org/amiede/classicthesis/