Evaluierung des Webentwicklungsframeworks eGovWDF im Hinblick auf die Aspekte Rich-Client-Funktionalität und Barrierefreiheit im Kontext der Anforderungen im Bereich eGovernment

Walter Kern

Lehrstuhl für Informationswissenschaft der Universität Regensburg

Juli 2009

Alle genannten und gegebenenfalls durch Dritte geschützten Marken- und Warenzeichen unterliegen uneingeschränkt den Bestimmungen des jeweils gültigen Kennzeichnungsrechts und den Besitzrechten der jeweiligen Eigentümer. Allein aufgrund der bloßen Nennung kann nicht geschlossen werden, dass Markenzeichen nicht durch Rechte Dritter geschützt sind. Aus dem Fehlen des Zeichens (R) darf nicht geschlossen werden, dass ein Name oder Zeichen frei ist. Eine Haftung für ein etwaiges Fehlen des Zeichens (R) wird ausgeschlossen.

Abstract

Wie im Rahmen einer umfangreichen Evaluation gezeigt wurde, erfüllt keines der derzeit erhältlichen Rich Client-Frameworks für Webanwendungen das sich im Schnittfeld aus Webentwicklung, Web 2.0-Technologien, modernem Software-Engineering und eGovernment ergebende Anforderungsprofil. Auf Basis des genannten Anforderungsprofils wurde im Rahmen der Realisierung eines übergeordneten Rich Client-Webentwicklungsframeworks (eGovWDF – eGovernmental Web Development Framework) für den Einsatz im eGovernment-Bereich ein barrierefreies Web 2.0-Komponentenframework realisiert, welches auf die vollständige Konformität zu diesen Anforderungen abzielt. Zielsetzung dieser Untersuchung ist nun die Prüfung des Grads der Anforderungserfüllung des eGovWDF-Teilframeworks bezüglich des in der oben genannten Evaluation spezifizierten Anforderungskatalogs.

Schlüsselbegriffe: Web 2.0, RIA, Rich Client, Webentwicklungsframework, Komponenten, Steuerelemente, eGovernment, Ajax, JavaScript-Unabhängigkeit

Abstract (en)

As shown by a preliminary evaluation, there is no rich client framework for web applications that suffices the requirements at the intersection of web development, Web 2.0, modern methods of software engineering and eGovernment. As a consequence, we extended our experimental rich client web development framework (eGovWDF - eGovernmental Web Development Framework) with a rich client components sub framework that aims at complete compliance with the requirements mentioned above. It is the objective of this evaluation to examine the degree of compliance of the eGovWDF sub framework with the catalog of requirements that was used in the preliminary evaluation.

Keywords: Web 2.0, RIA, Rich Client, web development framework, components, controls, eGovernment, Ajax, independence of JavaScript

Inhalt

1 Einleitung

Im Rahmen der Studie „Evaluierung bedeutender Webframeworks im Hinblick auf die Aspekte Rich-Client-Funktionalität und Barrierefreiheit im Kontext der Anforderungen im Bereich des eGovernment“ (vgl. Kern, 2008a) wurden die vielversprechendsten[1] und bedeutendsten[2] Webapplikationsframeworks im Hinblick auf die Erfüllung von Anforderungen aus der Domäne eGovernment im Schnittfeld der Primäraspekte Barrierefreiheit und Web 2.0 (vgl. Kern, 2008b; Kern, 2008c) realisierender Rich Client-Funktionalität untersucht.

Nach Kern (2008a) erfüllt keines der untersuchten Frameworks die vorliegenden Anforderungen in hinreichendem Maß, da selbst die drei Frameworks mit dem besten Testergebnissen im Hinblick auf den Gesamtanforderungserfüllungsgrad einen Grad an JavaScript-Unabhängigkeit von 0% besitzen, was im Hinblick auf den Aspekts Barrierefreiheit (vgl. Bundesministerium des Innern, 2002), als äußerst kritisch zu bewerten ist, da die auf den Frameworks basierenden Anwendungen größtenteils ohne JavaScript ihre Funktionalität einstellen. Diese Abhängigkeit von JavaScript ist laut Kern (2008a) bei allen untersuchten Rich Client-Frameworks gegeben und vor allem bei Einsatz von AJAX präsent; beispielsweise führt bei Verwendung des Frameworks Visual WebGui (vgl. Kern, 2008d) nahezu jede Nutzeraktion zu einem AJAX-Submit, was JavaScript voraussetzt. Bei Verschiebung des Untersuchungsfokus auf diejenigen Frameworks mit dem höchsten Grad an JavaScript-Unabhängigkeit ist festzustellen, dass insgesamt lediglich drei aller untersuchten Frameworks einen messbaren Grad an JavaScript-Unabhängigkeit besitzen. Dieser ist daran festzumachen, wieviele der im Rahmen der Untersuchung pro Framework getesteten Komponenten unabhängig von JavaScript-Verfügbarkeit auf Clientseite funktionieren. So konnte festgestellt werden, dass maximal eine 14%ige JavaScript-Unabhängigkeit pro Framework vorliegt, was bedeutet, dass jeweils eine von sieben getesteten Komponenten ihre Funktion bei fehlender JavaScript-Verfügbarkeit wahrt.

Aus diesem Grund wurde es in besagtem Dokument angeregt, ein Framework zu konzeptionieren, welches sich die Erfüllung der angeführten Anforderungen als Primärziel setzt. In diesem Zusammenhang ist eGovWDF[3] entstanden, dessen Rich Client-Komponenten-Teilframework versucht, einen möglichst hohen Grad an Anforderungsbefriedigung umzusetzen.

---

[1] Primärkriterium sind die Innovativität bzw. Grundsätzlichkeit eines Ansatzes, aber auch Wachstumspotentiale.

[2] Primärkriterien sind Marktdurchdringung des Produkts, Wachstumschancen und die Marktmacht des Herstellers.

[3] eGovWDF = eGovernmental Web Development Framework.

Zielsetzung dieser Arbeit ist damit nicht die funktionale und technische Beschreibung des genannten Teilframeworks von eGovWDF, sondern die Untersuchung der Anforderungserfüllung von eGovWDF im Hinblick auf die in Kern (2008a) formulierten Anforderungen, welche sich unter Anderem an generellen Forderungen an moderne flexible Software-Architekturen (vgl. Vogel et al., 2005; Boehm, 2006) orientieren, wenngleich der Primärfokus auf den Aspekten Barrierefreiheit im Kontext des (bundesdeutschen) eGovernment (vgl. Bundesministerium des Innern, 2002) und Rich Client-Funktionalitäten im Sinne von Rich Internet Applications (vgl. Moritz, 2008; Linaje et al., 2007; W3C, 2008) liegt.

Um die Einordnung der Ergebniswerte der Untersuchung des Rich Client-Komponentenframeworks von eGovWDF zu erleichtern, soll im Rahmen dieser Evaluation ein Vergleich mit den in Kern (2008a) untersuchten Frameworks stattfinden:

- ASP.NET 3.5 (Standard) – vgl. Esposito (2004); Esposito (2005); McClure et al. (2006)
- AJAX Control Toolkit – vgl. Wenz (2007)
- ComponentArt Web.UI – vgl. SYS-CON Media Inc. (2008)
- NetAdvantage for ASP.NET – vgl. Infragistics (2008a)
- Telerik Controls for ASP.NET AJAX – vgl. Telerik (2008)
- DevExpress ASPxperience – vgl. Developer Express Inc. (2008a); Developer Express Inc. (2008b)
- JSF (Standards) – vgl. Geary und Horstmann (2007)
- MyFaces Tomahawk – vgl. Marinschek et al. (2006)
- JBoss RichFaces – vgl. JBoss.org (2008a); JBoss.org (2008b)
- Oracle ADF Faces Rich Client – vgl. Oracle (2008)
- NetAdvantage for JSF – vgl. Infragistics (2008b)

2 Methodik der Evaluation

2.1 Hypothese und evaluierte Aspekte

Die über diese Evaluation zu belegende Hypothese besteht darin, dass das Rich Client-Komponenten-Teilframework für Webentwicklungen von eGovWDF, im späteren zur Vereinfachung kurz als eGovWDF bezeichnet, eine 100%-Anforderungserfüllung bezüglich der in Kern (2008a) formulierten Anforderungen und Komponententypen erreicht. Die in Kern (2008a) formulierten Anforderungen leiten sich dabei primär aus Verordnungen wie der BITV[4] (vgl. Bundesministerium des Innern, 2002) bzw. der BayBITV[5] (vgl. Bayerisches Staatsministerium des Innern, 2003), Dokumenten wie SAGA[6] (vgl. KBSt, 2006a) und der Forderung nach einer modernen flexiblen Software-Architektur (vgl. Vogel et al., 2005; Gamma et al., 1994), die sich auch in den Aspekten Generizität, Flexibilität und Erweiterbarkeit (vgl. KBSt, 2006b) wiederfindet, ab.

Die Untersuchung soll wie in Kern (2008a) Komponenten des Typs Dialog, Registerkarte, Menü, Symbolleiste, Baumsteuerelement, Fortschrittsanzeige und Infotip berücksichtigen, da diese Komponenten in Kern (2008a) als Schnittmenge verschiedener User Interface-Styleguides (vgl. Apple Inc., 2008; Benson et al., 2004; KDE-Entwicklerteam, o.J.; Microsoft Corporation, 2007a; Microsoft Corporation, 2007b) ermittelt worden sind.

Im Rahmen der Untersuchung soll äquivalent zu Kern (2008a) eine Gliederung der zu prüfenden Anforderungen in folgende Bereiche erfolgen, um ein optimales Maß an Komparabilität zu erreichen:

- Rich Client-Funktionalität
- Funktionsweise bei nicht verfügbarem JavaScript
- Barrierefreiheit nach BITV
- Browserinteroperabilität

Diesen thematischen (Anforderungs-)Bereichen sind in Kern (2008a) in Abhängigkeit vom Komponententyp unterschiedliche Detailanforderungen zugeordnet worden, die im Folgenden

[4] Barrierefreie Informationstechnikverordnung; vgl. http://www.gesetze-im-internet.de/bitv/BJNR265400002.html.

[5] Bayerische Barrierefreie Informationstechnikverordnung.

[6] SAGA = Standards und Architekturen für E-Government-Anwendungen.

aufgelistet werden. Hinsichtlich einer detaillierten Beschreibung dieser Aspekte soll auf Kern (2008a) verwiesen werden. Anzumerken ist, dass im Gegensatz zu Kern (2008a) und den darin vorgenommenen Framework-Dokumentations-gestützten Untersuchungen die Evaluierung von eGovWDF auf Grundlage des Expertenwissens des Architekten und Entwicklers beschrieben wird, weil aufgrund der Neuartigkeit von eGovWDF noch keine extensive schriftliche Dokumentation vorliegt. Gemeinsam ist der Evaluierung in Kern (2008a) und dieser Untersuchung die Verwendung von Framework-Funktionstests in Kombination mit einer visuellen Prüfung der sich bei Benutzung einer Frameworkkomponente ergebenden Weboberfläche sowie eine Betrachtung des resultierenden (X)HTML-Quellcodes durch einen Experten. Ebenso wird in beiden Evaluierungen die Client-Umgebung, insbesondere der Webbrowser des Clients, mit unterschiedlichem Funktionsumfang simuliert, indem bestimmte Funktionen testabhängig aktiviert bzw. deaktiviert werden. Exemplarisch sind die Deaktivierung von CSS, JavaScript und Farbe zu nennen, um die JavaScript-Unabhängigkeit sowie die Konformität zu den entsprechenden CSS- und Farbe-Unabhängigkeit fordernden BITV-Kriterien zu prüfen.

Komponententypabhängige Detailanforderungen im Bereich „Rich Client-Funktionalität“:

- Dialog
    - Request-Lebenszyklus-Persistenz
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
    - Wiederverwendbarkeit von Dialoginhalten
    - Definierter Datenaustauschprozess
    - Standard-Dialoge für MessageBoxen
- Registerkarten
    - Request-Lebenszyklus-Persistenz
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
- Menüs
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
- Symbolleisten

    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
- Hierarchische Steuerelemente
    - Request-Lebenszykus-Persistenz
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
- Fortschrittsanzeige
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung
- Infotip
    - Anpassbarkeit
    - Verhalten
    - Entwicklungsunterstützung

Neben diesen komponententypabhängigen Detailanforderungen im Bereich „Rich Client-Funktionalität“ stellen sich auch noch folgende übergreifende Anforderungen an die genannten Oberflächenkomponenten:

- Funktionsweise bei nicht verfügbarem JavaScript
- Barrierefreiheit nach BITV
- Browserinteroperabilität

Im Hinblick auf den Aspekt „Browserinteroperabilität“ ist dabei anzumerken, dass hierbei eine Unterstützung der folgenden, zum Zeitpunkt der Evaluierung von Kern (2008a) am häufigsten genutzten Browser (vgl. Net Applications, 2008) untersucht werden soll:

- Internet Explorer 6.0+
- Mozilla Firefox 2.0+
- Opera 9+
- Safari 3.1+

Um eine Vergleichbarkeit mit den Ergebnissen der Frameworkevaluierung nach Kern (2008a) zu erreichen, soll in dieser Evaluierung eine zur besagten Untersuchung identische Bewertung und

Gewichtung der oben genannten Anforderungen erfolgen. Daher wird für minimale Anforderungserfüllung ein Leistungswert von 0,0, für teilweise vorliegende Anforderungserfüllung ein Wert von 0,5 und für maximale Erfüllung ein Wert von 1,0 vergeben. Zusätzlich wird pro zu prüfendem Teilaspekt ein Gewichtungsfaktor spezifiziert, welcher über Produktbildung mit der Teilanforderungserfüllung in die Gesamtsummenbildung eingeht und Kern (2008a) entnommen wird.

2.2 Statistische Evaluationsmethoden

2.2.1 Methodenwahl

Der Grad der Gesamtanforderungserfüllung wird in dieser Evaluation identisch zu Kern (2008a) bestimmt, indem das gewichtete arithmetische Mittel pro Anforderungsbereich errechnet wird und von den jeweils vier resultierenden Werten ebenso das arithmetische Mittel ermittelt wird.

Grund für die Legitimität der Vereinfachung der Berechnung des Gesamterfüllungsgrads auf Basis der Formel für das arithmetische Mittel ist die Tatsache, dass der Maximalwert der Erfüllung pro Anforderung stets konstant 1 ist und dieser damit aus dem Nenner des Bruchs für die Gesamtanforderungsgrad-Bestimmung entfallen kann:

$$g_j = \frac{\sum_i r_i * a_{j,i}}{\sum_i r_i * m_i} \overset{m_i = 1 \,\forall\, i}{\Longrightarrow} \frac{\sum_i r_i * a_{j,i}}{\sum_i r_i} = \overline{a_j}$$

g_j : Grad der Gesamtanforderungserfüllung des Frameworks j

$\overline{a_j}$: Gewichtetes arithmetisches Mittel aller Anforderungserfüllungswerte des Frameworks j

$a_{j,i}$: Erfüllungswert der Anforderung i bei Framework j

m_i : Maximal möglicher Erfüllungswert der Anforderung i; konstant 1

r_i : Relevanzfaktor der Anforderung i (Gewichtungsfaktor)

Analog zu Kern (2008a) werden pro Komponentenframework neben dem Gesamtanforderungserfüllungsgrad die auf Teilmengen des Gesamtanforderungskatalogs beruhenden Kenngrößen JS-Unabhängigkeit und Komponentenabdeckungsgrad ermittelt.

Um die qualitativen sowie quantitativen Unterschiede zwischen dem eGovWDF-Ansatz und den geprüften Lösungen des aktuellen Stands der Technik im Detail, d.h. bezogen auf die einzelnen Anforderungen und bezogen auf die Anforderungsbereiche zu analysieren, wird im Rahmen dieser

Untersuchung zusätzlich eine Berechnung von Durchschnitt, Median, unterem und oberem Quartil sowie den Extremwerten vorgenommen.

Beachtet werden sollte hierbei, dass sämtliche der oben genannten Kennzahlen - unabhängig von eGovWDF - für die in Kern (2008a) evaluierten Frameworks errechnet werden, da ansonsten eine Verfälschung der Kennzahlen auftritt. Beispielsweise würde der Wert der durchschnittlichen Anforderungserfüllung, der die aktuelle Situation ohne eGovWDF beschreiben soll, durch eine Berücksichtigung der Kennzahlen von eGovWDF verzerrt.

2.2.2 Begründung der Methodenwahl

Aufgrund der Primärzielsetzung des Aufzeigens des aktuellen Stands der Technik und der aktuellen Maximal-Anforderungserfüllung bei gegenwärtig verfügbaren Rich Client-Komponentenframeworks, wurde in Kern (2008a) eine Beschränkung auf die Betrachtung der Maximal- und der Durchschnittsanforderungserfüllung der untersuchten Frameworks vorgenommen. Zudem wurde die JS-Unabhängigkeit pro Komponentenframework ermittelt, weil diese von zentraler Relevanz ist, da die Barrierefreiheit als nicht gegeben zu bewerten ist, falls diese essentielle Anforderung vom jeweiligen Framework nicht unterstützt wird, und Barrierefreiheit im eGovernment-Umfeld eine Pflichtanforderung ist. Ferner wird der Komponentenabdeckungsgrad ermittelt, welcher angibt, wieviel Prozent der im Rahmen der Evaluierung von Kern (2008a) geforderten Komponenten vom jeweiligen Framework unterstützt werden. Grundsätzlich werden auch die in Kern (2008a) angesprochenen Gewichtsfaktoren berücksichtigt, deren Multiplikation mit den Erfüllungsgradwerten der zugehörigen Anforderungen summarisch in die Berechnung der Gesamtanforderung einfließt. Aufgrund der Äquivalenz aller Gewichtungsfaktoren mit dem Wert 1, ist jedoch kein Einfluss dieser Größen auf den Gesamterfüllungsgrad vorliegend. Der Grad der Gesamtanforderungserfüllung wird in dieser Evaluation damit identisch zu Kern (2008a) bestimmt, indem das gewichtete arithmetische Mittel pro Anforderungsbereich errechnet wird und von den jeweils vier resultierenden Werten ebenso wieder das arithmetische Mittel berechnet wird. Unabhängig davon ergeben sich für die Wahl der gewählten Evaluationsmethoden die im Folgenden genannten Gründe.

Die Gesamtanforderungserfüllung pro Framework wird grundsätzlich ermittelt, um eine Maßzahl für die Anwendbarkeit des jeweiligen Frameworks bei konkreten Projekten in der sich aus den Einzelanforderungen ergebenden Problemdomäne zu erhalten. Diese skalare Maßzahl ermöglicht

damit auch die Ermittlung einer Framework-Rangfolge und allgemein den gesamtheitlichen Vergleich mehrerer Frameworks.

Über die Maßzahl der Durchschnittsanforderungserfülllung und damit des gewichteteten arithmetischen Mittels über die Gesamtanforderungserfüllungswerte aller Frameworks soll festgehalten werden, welcher Gesamtanforderungserfüllungsgrad im Mittel erreicht wird, wodurch eine Aussage über den bei einer Schnittfeldbetrachtung der Frameworks sich ergebende mittlere Anforderungserfüllung gegeben wird, was wiederum als Vergleichsgröße gegenüber den Gesamtanforderungserfüllungswerten der einzelnen Frameworks dienen kann.

Da das gewichtete arithmetische Mittel sehr anfällig für Extremwerte ist, eignet es sich nicht für die Bestimmung des Stands der Technik, sodass zusätzlich eine Berechnung des Medianwerts erfolgt, welcher als wesentlich störresistenter gilt und es in Kombination mit Minimum- und Maximumwerten erlaubt, starke Ausreißer nach oben (potentielle Lösungen über dem Stand der Technik) und unten klar davon abzugrenzen.

Ferner sollen oberes und unteres Quartil berechnet werden, um eine Einteilung in gesamtheitlich „gute Frameworks“ und „schlechte Frameworks“ vornehmen zu können.

Um neben der globalen eine feingranulare Bewertung der Frameworks zu erhalten, werden die genannten Kennzahlen in der Untersuchung jeweils auch auf die zu Anforderungsarten bzw. Komponenten gruppierten Teilmengen des Gesamtanforderungskatalogs angewendet.

2.3 Visuelle Evaluationsmethoden

Zur Visualisierung der Kennzahlen Maximal- und Durchschnittsanforderungserfüllung wurde angelehnt an Kern (2008a) eine Balken- bzw. Säulendiagrammdarstellung, welche die prozentuale Gesamt-Anforderungserfüllung jedes getesteten Frameworks aufzeigt, als probates Mittel bestimmt. Damit wird auch die Komparabilität sichergestellt, da in Kern (2008a) die identische Darstellungsform gewählt wurde und somit ein Überblicks-Vergleich ermöglicht wird.

Zur weitergehenden Analyse ist die Entscheidung für eine Boxplot-Visualisierung (vgl. Tukey, 1977) gefallen, da diese es erlaubt, alle unter 2.2 beschriebenen Kenngrößen in einer Darstellung kompakt und die Komparabilität fördernd, zu visualisieren. Sowohl pro

Gesamtanforderungserfüllungswert als auch pro Bereichsanforderungserfüllungswert wird der Boxplot-Visualisierung eine metrische Skala, genauer eine Intervallskala, zugrunde gelegt, da ein direkter Vergleich der erreichten Anforderungserfüllung möglich ist.

3 Evaluierung

3.1 Beschreibung

Bei eGovWDF handelt es sich um ein Framework mit der Zielsetzung die besonderen Anforderungen und Belange in der Domäne eGovernment zu realisieren. Dabei besteht eGovWDF aus verschiedenen Teilframeworks um die verschiedenen Themenfelder in diesem Umfeld abzubilden, wozu ein Validierungsframework, ein barrierefreies Rich-Client-Framework und ein Teilframework für assistive Benutzerführung zählen (Abbildung 1).

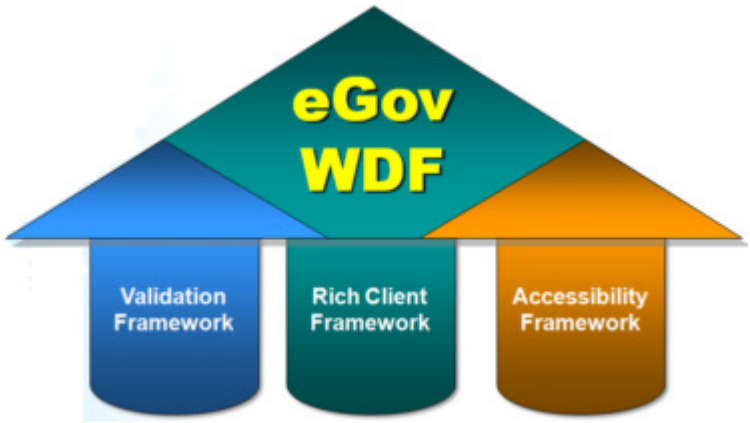


Abbildung 1: Teilframeworks von eGovWDF.

Das barrierefreie Rich Client-Framework von eGovWDF wurde auf Basis der in Kern (2008a) dargestellten Anforderungen realisiert, sodass ein hoher Grad an Anforderungserfüllung vermutet wird, was im Rahmen dieser Arbeit nun untersucht werden soll. Es zielt darauf ab die gegenwärtig vorliegende Unvereinbarkeit von Anforderungen aus dem Bereich Rich Client-Funktionalität mit Anforderungen aus dem Bereich Barrierefreiheit zu überwinden (vgl. Kern, 2008b; Kern, 2008c, Hailpern et al., 2009).

3.2 Evaluierte Produktversion

Gegenstand dieser Untersuchung ist Version 1.0 der ASP.NET-Referenzimplementierung des Rich Client-Teilframeworks von eGovWDF.

3.3 Detailuntersuchung

3.3.1 Dialoge

3.3.1.1 Beschreibung

eGovWDF unterstützt die Realisierung von Dialogen durch die Kombination der zwei Komponenten WebDialogReference und WebDialog.

Zur Erreichung maximaler Wiederverwendbarkeit werden Dialoge als separate Ressourcen angelegt und deren Inhalt nicht direkt als Bestandteil der jeweiligen Webseite eingebettet. Aus diesem Grund wird ein Dialog als neue Webressource ähnlich einer neuen Webseite angelegt, wobei der Dialog-Inhalt über die herkömmlichen Visual Studio 2008-Designer für Webseiten (WebPages) befüllt werden kann. Über die WebDialog-Eigenschaften können die Anzeige- und Verhaltenparameter des jeweiligen Dialogs festgelegt werden, beispielsweise Titelleistenbeschriftung, Standard-Buttons und Drag & Drop-Unterstützung.

Auf der jeweiligen Webseite, von der aus ein Dialog aufgerufen werden soll, ist eine entsprechende WebDialogReference-Komponente zu positionieren, wobei der relative oder absolute Pfad zur Dialog-Ressource konfiguriert werden muss. Sämtliche Verhaltens- und Anzeigeeigenschaften werden von der WebDialog-Ressource übernommen, können aber komfortabel über die WebDialogReference-Eigenschaften lokal überschrieben werden.

Das Festlegen der Schaltflächen zum Ein- und Ausblenden eines Dialogs erfolgt rein deklarativ und kann über einen Designer komfortabel festgelegt werden, sodass keinerlei Code zur Anzeige bzw. zum Schließen eines Dialogs geschrieben werden muss. Diese deklarative Festlegung ermöglicht dem Framework zudem die verschiedenen Modi der Dialogaktivitäten umzusetzen. So kann ein Dialog bereits zum Zeitpunkt des Ladens der aufrufenden Webseite unsichtbar gerendert werden und dann über JavaScript ein- oder ausgeblendet werden. Alternativ kann der Dialog bei der Betätigung eines Triggers zur Anzeige on demand nachgeladen und gerendert werden. Schließlich besteht auch die Möglichkeit das Ein- und Ausblenden des Dialogs rein serverseitig vorzunehmen. Auch sind verschiedene Kombinationen der genannten Optionen denkbar, wie beispielsweise das Laden eines Dialogs beim ersten Zugriff und das zukünftige Ein- und Ausblenden über JavaScript, wobei bei Nicht-Verfügbarkeit von JavaScript das initiale Laden und die späteren Ein- und Ausblendaktivitäten rein serverseitig erfolgen, wodurch auch dem Aspekt

Barrierefreiheit aufgrund der Verzichtbarkeit auf JavaScript ohne Funktionsdefizite entsprochen wird. Zusammenfassend ermöglicht die deklarative Festlegung, dass vom Entwickler kein entsprechender Code für die clientseitige und/oder serverseitige Anzeige bzw. das Ausblenden sowie das dynamische Nachladen geschrieben werden muss, da dies transparent im Hintergrund erfolgen kann. Der scheinbare Nachteil der fehlenden Flexibilität zur bedingten Anzeige bzw. der Ausblendung von Dialogen wird durch client- und serverseitige Ereignisbehandlungsroutinen, die während der Dialoglebenszeit aufgerufen werden kompensiert, da innerhalb dieser Routinen gesteuert werden kann, ob der Dialog tatsächlich ein- bzw. ausgeblendet werden soll und unter welchen Bedingungen.

Im Hinblick auf die Datenübergabe zwischen Dialog und Dialogaufrufer wird ein sogenannter Dialog-Kontext bereitgestellt, über welchen Daten bidirektional ausgetauscht werden können. Der Austausch wird dabei vom Framework im Hintergrund sowohl bei AJAX-Verfügbarkeit als auch bei Nicht-Verfügbarkeit rein serverseitig transparent umgesetzt.

Ferner soll erwähnt werden, dass weitere Komfortfunktionalität wie Dialogmodalität und die Unterstützung von Standardtastenkombinationen unterstützt und dialogbezogen bzw. auf Triggerebene konfiguriert werden können.

In Abbildung 2 wird exemplarisch ein Dialog zur Auswahl eines Navigationsziels dargestellt.

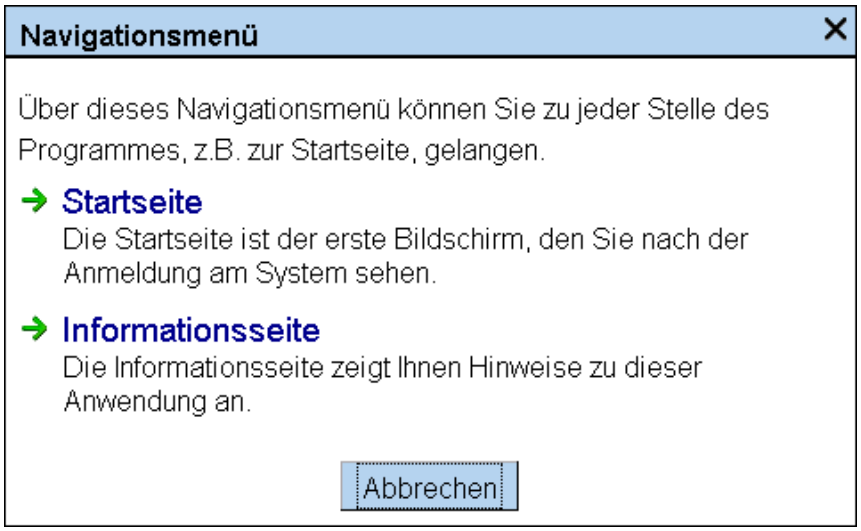


Abbildung 2: Das eGovWDF-WebDialog-Steuerelement.

3.3.1.2 Rich-Client-Funktionalität

Request-Lebenszyklus-Persistenz

Der Sichtbarkeitszustand des Dialogs wird über Requestgrenzen hinweg persistiert.

Bewertung: 1,0

Anpassbarkeit

Der Dialog-Inhalt kann völlig frei angepasst werden. Es können beliebige HTML- oder Serverkomponenten innerhalb eines Dialogs positioniert werden. Die Titelleiste kann flexibel über CSS sowie weitere Eigenschaften modifiziert werden. Der Dialog sowie dessen Bestandteile unterstützen ASP.NET-Theming-Mechanismen um Design-Festlegungen auf Steuerelementebene vorzunehmen. Auch weitere Anpassungen wie beispielsweise die Einblendung eines Icons für Direkthilfe werden unterstützt. Auch besteht die Möglichkeit die Titelleiste über CSS auszublenden und eine benutzerdefinierte Titelleiste festzulegen. Sämtliche Dialog-Bestandteile sind ferner über CSS anpassbar, inklusive des Schließen-Symbols und der Mausanzeige während eines Dragvorgangs.

Bewertung: 1,0

Verhalten

Es kann eine beliebige Anzahl von Komponenten als Trigger zur Dialoganzeige bzw. zum Ausblenden eines Dialogs deklarativ und designerunterstützt festgelegt werden. Zusätzlich unterstützt die WebDialog-Komponente Funktionalitäten wie Drag & Drop und Standardtastenkombinationen. Auch wird das dynamische Nachladen von Dialoginhalten on demand sowie eine Graceful Degradation ohne Verlust der Kernfunktionalität bei Nicht-JavaScript-Verfügbarkeit unterstützt.

Bewertung: 1,0

Entwicklungsunterstützung

Eine gute Unterstützung liegt bei Benutzung von Visual Studio 2008 vor. Es wird eine neue Elementvorlage in Visual Studio bereitgestellt über welche anstatt von WebPages WebDialoge erzeugt werden können. Die Befüllung eines WebDialogs erfolgt analog zur Befüllung einer WebPage über den Standard-Visual-Studio-Designer. WebDialog-Eigenschaften können auch

programmatisch festgelegt werden. Zudem können sämtliche WebDialogReference-Eigenschaften über das Visual Studio 2008-Eigenschaftsfenster gepflegt werden.

Bewertung: 1,0

Wiederverwendbarkeit von Dialoginhalten

Die Aspekte Dialogdefinition und Dialogverwendung sind durch die Aufsplittung auf die zwei Komponenten WebDialog und WebDialogReference voneinander separiert, sodass ein WebDialog von beliebig vielen WebPage-Ressourcen referenziert und verwendet werden kann.

Bewertung: 1,0

Definierter Datenaustauschprozess

Es existiert eine definierte, dialogimplementierungsoriginäre Möglichkeit um Informationen zwischen dem Dialog-Owner, also der öffnenden Webseite, und dem Dialog auszutauschen. Diese wird als DialogContext bezeichnet und erlaubt das übergeben beliebiger, auch benutzerdefinierter, .NET-Objekte über einen HashMap-basierten Mechanismus. Aufgrund der deklarativen Trigger-Definition, erfolgt die Übergabe transparent über AJAX oder alternativ rein serverseitig, falls AJAX nicht verfügbar ist bzw. eine entsprechende Konfiguration erfolgt ist. Ferner wurde eine analoge clientseitige API im Hinbick auf diese Funktionalität umgesetzt, sodass (aus Performance-Gründen) eine rein clientseitige Lösung zusätzlich zur serverseitigen bzw. AJAX-basierten Realisierung möglich ist.

Bewertung: 1,0

Standard-Dialoge für MessageBoxen

Standard-Dialoge für MessageBoxen werden über das Komponentenduo WebMessageBox und WebMessageBoxReference realisiert. Diese leiten sich dabei von den entsprechenden WebDialog- und WebDialogReference-Gegenstücken ab und erlauben die Anzeige aller Dialog-Arten, wie sie auch bei herkömmlichen Windows-Dialogen unterstützt werden. Im Detail kann das Dialog-Icon festgelegt werden und es werden folgende Schaltflächen-Kombinationen unterstützt:

- Abbrechen, Wiederholen, Ignorieren
- OK
- OK, Abbrechen

- Wiederholen, Abbrechen
- Ja, Nein
- Ja, Nein, Abbrechen

Ein Aufruf zur Anzeige eines Dialogs wird wie bei den Dialogen nicht über imperative Programmierung, sondern über deklarative Trigger-Zuordnungen realisiert. Analog zu den Dialogen werden verschiedene Dialog-Rückgabewerte unterstützt. Dabei handelt es sich per Default um folgende Typen:

- None
- Accept
- Decline
- Cancel

Zusätzlich können beliebige benutzerdefinierte Rückgabewerte über zusätzliche DialogContext-Parameter zurückgegeben werden.

Bewertung: 1,0

3.3.1.3 Funktionsweise bei nicht-verfügbarem JavaScript

Die Deaktivierung von JavaScript hat keinen Einfluss auf die Funktionalität des Dialogs mit Ausnahme der Standardtastenkombinationen und der Drag & Drop-Funktionalität, die aber ohnehin lediglich optionalen und nicht funktionszentralen Charakter aufweisen. Zudem wird bei Nicht-Verfügbarkeit von JavaScript eine barrierefreie Alternativmöglichkeit zur Repositionierung bzw. Größenänderung von Dialogen unterstützt. Sämtliche Primär-Funktionalität wie das Ein- und Ausblenden, das dynamische Nachladen von Inhalt sowie der Datenaustausch zwischen Aufrufer und Dialog sind unabhängig von JavaScript verfügbar.

Bewertung: 1,0

3.3.1.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Daneben verwendet keine Komponente zwingend Symbole. Falls welche eingesetzt werden, z.B. Hilfe- oder Schließenicons in der Dialogtitelleiste wird für diese die Spezifikation von Alternativtexten vom Framework erzwungen, sodass ohne Spezifikation dieser im Sinne der Barrierefreiheit relevanten Attribute die Anwendung nicht ausgeführt wird. Zudem werden keine Image-Maps oder Multimedia-Präsentationen verwendet.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponente im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass für alle relevanten Bestandteile des Dialogs (Header- und Inhaltsbereich) eine Anpassung über CSS-Klassen und die Standard-ASP.NET-Theming-Mechanismen unterstützt wird. Zur Befüllung der Webdialoge besteht umfangreiche Visual Studio-Designer-Unterstützung. Es wird XHTML Strict 1.0-Code und standardkonformes CSS verwendet. Dasselbe trifft auf die JavaScript-Konformität zu.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung der Dialog-Komponenten vorliegt. Die Festlegung von

Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von den Dialog-Steuerelementen zu sehen.

Bewertung: 1,0

Anforderung 5

Keine der Dialog-Komponenten basiert auf Tabellen, weshalb keine Anwendbarkeit dieses Prüfaspekts vorliegt.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS ist die Anwendung grundsätzlich noch verwendbar, jedoch schwerer bedienbar, da kein Unterschied mehr zwischen angezeigten und verborgenen Dialogen existiert, falls die entsprechenden Optionen entsprechend konfiguriert sind. Es kann jedoch auch eingestellt werden, dass der jeweilige Dialog stets nur in den HTML-Output gerendert wird, falls dieser sichtbar ist, wodurch bei nicht verfügbarem CSS keine Anzeige und bei verfügbarem CSS eine Anzeige des Dialogs erfolgt. Ein Deaktivieren von JavaScript hat keinerlei Einfluss auf die Kernfunktionalität eines Dialogs.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext der Dialogsteuerelemente nicht anwendbar, da keine zeitgesteuerten Funktionalitäten bis auf das dynamische Laden des Dialogs on demand via AJAX vorliegen. Für diese Operation besteht jedoch über das ProgressPresenter-Steuerelement des Frameworks die Möglichkeit eine Fortschrittsanzeige zu realisieren. Zudem werden Screenreader-Benachrichtigungen bei Ein- und Ausblendung von Dialogen unterstützt.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt. Es werden jedoch explizit konfigurierbare Screenreader-Benachrichtigungen bei komponentenbezogenen, bildschirmaktualisierenden Aktivitäten unterstützt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine schlüssige Tabulatorreihenfolge ist standardmäßig gegeben. Tastaturkürzel (Accesskeys) können für die Dialogbestandteile wie das Schließen-Symbol sowie auch unabhängig vom Dialog für die enthaltenen Kindelemente vergeben werden. Sämtliche Steuerelemente, die zur Anzeige bzw. zum Ausblenden eines Dialogs unterstützt werden und wählbar sind, sind Komponenten, die bei fehlender JavaScript-Verfügbarkeit einen Submit der kompletten Webseite zur Folge haben und damit eine serverseitige Verarbeitung gestatten.

Bewertung: 1,0

Anforderung 10

Es werden keine regulären Popups verwendet. Stattdessen handelt es sich bei den Dialog-Komponenten um Bereiche der jeweiligen Webseite, die dynamisch ein- und ausgeblendet werden.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die Dialog-Steuerelemente selbst machen von keinerlei Frames Gebrauch. Zur verbesserten Orientierung und Navigation werden spezifische Elemente, wie z.B. Titelleisten sowie die

grundsätzliche Struktur von Dialogen bereitgestellt. Zur Realisierung von MessageBoxen existiert ein separates Komponentenduo bestehend aus WebMessageBox und WebMessageBoxReference.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext der Dialog-Steuerelemente nicht relevant. Die Navigationsoptionen im Dialogkontext werden über alternative Texte, z.B. im Hinblick auf das Schließen-Symbol, erläutert. Zudem werden Screenreader-Benachrichtungen bei Dialogaktivitäten unterstützt.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte der Dialog-Komponenten liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.1.5 Browserinteroperabilität

Die Dialog-Komponenten funktionieren unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.2 Registerkarten

3.3.2.1 Beschreibung

eGovWDF unterstützt die Realisierung von Registerkarten durch das TabControl-Steuerelement, welches in Abbildung 3 exemplarisch visualisiert wird.

Abbildung 3: Das eGovWDF-TabControl-Steuerelement.

3.3.2.2 Rich-Client-Funktionalität

Request-Lebenszyklus-Persistenz

Der Zustand der Registerkarten-Komponente wird über Request-Grenzen hinweg persistiert, unabhängig von der Verfügbarkeit von JavaScript. Auch ein Wechsel zwischen JavaScript-Verfügbarkeit und JavaScript-Nicht-Verfügbarkeit zwischen zwei Requests wird korrekt verarbeitet. Clientseitige Zustandsänderungen werden für die Verarbeitung auf der Serverseite persistiert und beim nächsten Request inklusive ggf. auf der Serverseite durchgeführten Änderungen zum Client zurückübermittelt. Auch ein Lazy Loading von Registerkarteninhalten beim ersten Abruf wird unterstützt und korrekt persistiert.

Bewertung: 1,0

Anpassbarkeit

Der Registerkarteninhalt lässt sich völlig frei anpassen. Die Registerkartenknöpfe sind über CSS und über diverse Eigenschaften anpassbar. So können nicht nur einfache Beschriftungen, sondern auch beliebige Symbole dargestellt werden. Auch kann festgelegt werden, ob die Registerkarten an der Ober- oder Unterseite des Registerkartensteuerelements angezeigt werden sollen. Ferner ist über CSS und benutzerdefinierte Submitelemente die Bereitstellung von ersatzweisen Registerkartenknöpfen möglich, um maximale Flexibilität zu erhalten. Es werden jedoch keine anderen Steuerelemente für Button-Knöpfe als Submit- oder Image-Submit-Buttons unterstützt, da nur diese bei Nicht-JavaScript-Verfügbarkeit einen Submit auslösen und damit eine serverseitige Verarbeitung sicherstellen.

Bewertung: 1,0

Verhalten

Ein Klick auf einen Registerkartenknopf bewirkt die Anzeige der zugehörigen Registerkarte. Eine reine Bedienung mit der Tastatur ist über die Tab- und Entertaste oder auch über frei festlegbare Access-Keys browserübergreifend möglich. Die Bedienung des Registerkartensteuerelements funktioniert unabhängig von JavaScript.

Bewertung: 1,0

Entwicklungsunterstützung

Eine gute Unterstützung liegt bei Benutzung von Visual Studio 2008 vor. Registerkarten eines Registerkartensteuerelements können via Designer komfortabel hinzugefügt werden. Auch ist eine Live-Umschaltung zwischen den einzelnen Registerkarten auf der Designeroberfläche per Klick auf den jeweiligen Tab und eine Befüllung via Drag & Drop möglich. Daneben besteht SmartTag-Unterstützung.

Bewertung: 1,0

3.3.2.3 Funktionsweise bei nicht-verfügbarem JavaScript

Die Deaktivierung von JavaScript hat keinen Einfluss auf die Funktionalität des Registerkartensteuerelements. Der Registerkartenwechsel kann bei deaktiviertem JavaScript unverändert über eine Kombination von Tab- und Enter-Taste bzw. AccessKeys oder alternativ per Klick auf die Registerkarten durchgeführt werden. Auch stehen erweiterte Funktionen wie Lazy Loading weiterhin zur Verfügung, wobei diese dann statt via AJAX über einen vollständigen Submit erfolgen.

Bewertung: 1,0

3.3.2.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Daneben verwendet keine Komponente standardmäßig Symbole. Für die Registerkartenknöpfe werden jedoch Symbole unterstützt, wobei die Festlegung von Alterantivtexten vorgesehen ist. Zudem werden keine Image-Maps oder Multimedia-Präsentationen verwendet.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponente im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass sowohl für die Komponente in ihrer Gesamtheit als auch für jeden einzelnen Tab die CSS-Klasse festgelegt werden kann. Daneben können die Anordnung der Registerkarten sowie die Anzeige der Registerkartenknöpfe im Hinblick auf Text- und/oder

Symboldarstellung konfiguriert werden. Der Inhalt einer Registerkarte kann zudem völlig frei vorgegeben werden, was auch für die zugehörige CSS-Klasse gilt. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung der Tab-Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von der Tab-Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

Tab-Komponenten basieren auf div- und span-Elementen und nicht auf Tabellen, weshalb keine missbräuchliche Verwendung von Tabellen vorliegt.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS ist die Anwendung weiterhin benutzbar. Alle Registerkarteninhalte werden aber dann standardmäßig ständig und untereinander dargestellt. Falls entsprechend konfiguriert, wird jedoch immer nur der Inhalt des aktuell gewählten Tabs in den HTML-Ausgabestrom gerendert, wodurch bei deaktiviertem JavaScript korrekterweise nur die gewählten Registerkarten sichtbar sind. Eine Deaktivierung von JavaScript hat keinen Einfluss auf das Funktionieren der Komponente.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext der Registerkartensteuerelemente nicht anwendbar, da keine zeitgesteuerten Funktionalitäten bis auf das dynamische Laden von Registerkarteninhalten on demand via AJAX vorliegen. Für diese Operation besteht jedoch über das ProgressPresenter-

Steuerelement des Frameworks die Möglichkeit eine Fortschrittsanzeige zu realisieren. Zudem werden Screenreader-Benachrichtigungen bei Ein- und Ausblendung von Registerkarten unterstützt.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt. Es werden jedoch explizit konfigurierbare Screenreader-Benachrichtigungen bei komponentenbezogenen, bildschirmaktualisierenden Aktivitäten unterstützt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine schlüssige Tabulatorreihenfolge innerhalb einer Registerkarte ist standardmäßig gegeben. Die Registerkartenknöpfe können in allen getesten meisten Browsern fokussiert werden und ein Registerkartenwechsel ist damit über die Kombination von Tab- und Entertaste, über Access Keys sowie per Mausklick durchführbar.

Bewertung: 1,0

Anforderung 10

Die Registerkartenkomponente setzt keine Popups ein.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die Registerkarten-Steuerelemente selbst machen von keinerlei Frames Gebrauch. Zur verbesserten Orientierung und Navigation werden Submit-Elemente als Registerkartenknöpfe verwendet, welche ein direktes Anspringen über die Tabtaste und eine Auswahl über die Entertaste ermöglichen. Dies ermöglicht es auch unabhängig von JavaScript eine Auswahl der Navigationsmöglichkeiten über die Kombination aus Tab- und Entertaste, über Accesskeys oder per Mausklick vorzunehmen.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext dieser Steuerelemente nicht relevant. Die Navigationsoptionen werden über alternative Texte, z.B. im Hinblick auf die Registerkartenknöpfe, erläutert. Zudem werden Screenreader-Benachrichtungen bei Registerkartenaktivitäten unterstützt.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.2.5 Browserinteroperabilität

Das Registerkartensteuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.3 Menüs

3.3.3.1 Beschreibung

Menüs können über das MenuControl-Steuerelement (vgl. Abbildung 4) sowie die untergeordneten Komponenten Menu und MenuItem realisiert werden. Ein MenuControl enthält beliebig viele Menu-Komponenten, womit eine Art Hauptmenü realisiert wird. Dem Hauptmenü können über MenuItem-Komponenten weitere Untermenüs hinzugefügt werden. Es wird damit ein zweistufiges Menüsystem unterstützt. Die entsprechende Einschränkung in der Verschachtlungstiefe wurde bewusst gewählt, da tief verschachtelte Strukturen einen negativen Einfluss auf Usability und Accessibility aufgrund der Zunahme an Komplexität besitzen können. Ein weiterer Grund ist die Erreichung von barrierefreier, JavaScript-unabhängiger Browserinteroperabilität. Falls die Erfordernis für weitere Untergliederungen besteht, werden Navigationsdialoge unterstützt, die zusätzlich detaillierte Hilfebeschreibungen zu den unterstützten Menüoptionen darstellen. In Abbildung 4 wird exemplarisch ein über die genannten Komponenten realisiertes Menü visualisiert.

Abbildung 4: Das eGovWDF-MenuControl-Steuerelement.

3.3.3.2 Rich-Client-Funktionalität

Anpassbarkeit

Das Menu-Steuerelement ist flexibel anpassbar. So können via CSS alle Menü-Einträge auf beliebiger Ebene über globale Style-Klassen in der Darstellung festgelegt werden. Darüber hinaus ist es möglich, einem Menüelement neben Text auch Symbole hinzuzufügen, wobei für alle Elemente alternative Texte spezifizierbar sind. Damit können beliebige Submit-auslösende Komponenten als Menüelement eingesetzt werden, wobei andere Elemente bewusst ausgeschlossen werden, da nur Submitelemente bei Nicht-JS-Verfügbarkeit eine serverseitige Verarbeitung ermöglichen.

Bewertung: 1,0

Verhalten

Ein Bewegen der Maus auf ein Menu-Element bewirkt die Anzeige der untergeordneten Menüelemente, welche bei Mausklick darauf eine Navigation oder die Verarbeitung eines serverseitigen Ereignisses erlauben. Das Ein- und Ausblenden der Menüelemente funktioniert unabhängig von JavaScript, da CSS eingesetzt wird. Die Menüelement-Befehlsausführung funktioniert unabhängig von JavaScript, da echte Submit-Events bei Nicht-JS-Verfügbarkeit ausgelöst werden.

Bewertung: 1,0

Entwicklungsunterstützung

Der Aufbau der Menüstrukturen wird durch den sogenannten Menu-Auflistungs-Editor und den MeuItem-Auflistungs-Editor auf einfache Art und Weise ermöglicht.

Bewertung: 1,0

3.3.3.3 Funktionsweise bei nicht-verfügbarem JavaScript

Die Menükomponente verwendet kein JavaScript, sodass die Funktionalität unabhängig von JavaScript gewahrt ist.

Bewertung: 1,0

3.3.3.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Bei Spezifikation von Menüsymbolen können Alternativtexte sowie Metainformationen über Tooltips spezifiziert werden.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keine negative Bewertung der Komponenten im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass für die Komponente global diverse graphische Einstellungen vorgenommen werden können, inklusive auf die verschiedenen Elemente anzuwendende Vorder- und Hintergrundfarben. Die Darstellungseigenschaften können zudem auf Elementebene überschrieben werden. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung dieser Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von dieser Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

Es findet keine Zweckentfremdung des table-Elements bei dieser Komponente statt, da für die Darstellung lediglich eine Verschachtelung von span-Elementen stattfindet. Menüelemente selbst werden aus Gründen der Barrierefreiheit (Funktionsweise ohne JavaScript) als Submitelemente dargestellt.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS ist die Anwendung weiterhin benutzbar. Jedoch werden dann alle Menüelemente unter- bzw. nebeneinander dargestellt. Es findet damit kein explizites Einblenden von Submenüelementen mehr statt, da stets alle sichtbar sind. Die Funktionalität bei Betätigung eines Menüeintrags wird jedoch nicht beeinflusst und steht unabhängig von JavaScript- und/oder CSS-Verfügbarkeit uneingeschränkt zur Verfügung.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext des evaluierten Steuerelements nicht anwendbar, da keine zeitgesteuerten Funktionalitäten vorliegen.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine Ansteuerung mit der Tastatur ist möglich.

Bewertung: 1,0

Anforderung 10

Diese Komponenten setzen keine Popups ein.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die Menü-Steuerelemente selbst machen von keinerlei Frames Gebrauch. Die Struktur des Menüs wird über span-Elemente realisiert, wobei die Menüelemente selbst durch Submitelemente realisiert worden sind, was eine einfache JavaScript-unabhängige Selektion der Navigationsoptionen über die Tastatur gestattet.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext dieser Steuerelemente nicht relevant. Die Struktur des Menüs wird über span-Elemente realisiert, wobei die Menüelemente selbst durch Submitelemente realisiert

worden sind, was eine einfache JavaScript-unabhängige Selektion der Navigationsoptionen über die Tastatur gestattet. Die Metainformation über die Elemente, welche die Navigation ermöglichen, ist darin begründet, dass alle navigationsauslösenden Menüelemente über Submitsteuerelemente realisiert werden. Sprungziele bei Hyperlinks sind neben der textuellen Beschreibung auch über Tooltips illustrierbar. Zusammengehörige Elemente können über Separatoren auch visuell als zusammengehörig gekennzeichnet werden.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.3.5 Browserinteroperabilität

Das jeweilige Steuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.4 Symbolleisten

3.3.4.1 Beschreibung

Symbolleisten können über das Steuerelement ToolBarControl (vgl. Abbildung 5) realisiert werden. Dabei werden als ToolBar-Komponenten Schaltflächen, Textfelder, Dropdownmenüs und Separatoren unterstützt. Für die einzelnen Symbolleistenelementtypen werden wiederum verschiedene Darstellungsoptionen unterstützt, z.B. bei Schaltflächen eine Text- und/oder Bilddarstellung.

Abbildung 5: Das eGovWDF-ToolbarControl-Steuerelement.

3.3.4.2 Rich-Client-Funktionalität

Anpassbarkeit

Das ToolBar-Control unterstützt die oben genannten Steuerelementtypen. Damit ist der Großteil der Anwendungsszenarien abgedeckt. Zudem besteht die Möglichkeit weitere Sub-Komponenten über einen Adaptermechanismus und Vererbung zu unterstützen. Sämtliche Elemente können über CSS bzw. konkrete Komponenteneinstellungen flexibel angepasst werden.

Bewertung: 1,0

Verhalten

Eine Bewegung des Mauszeigers zeigt das entsprechende Symbolleistenelement im festgelegten Hover-Zustand an. Ein Klick darauf führt die hinterlegte Aktion unabhängig von der Verfügbarkeit von JavaScript aus.

Bewertung: 1,0

Entwicklungsunterstützung

Symbolleistenkomponenten können komfortabel über den ToolbarItem-Auflistungs-Editor konfiguriert werden, was das Hinzufügen, Bearbeiten und Löschen von Symbolleistenelementen miteinschließt.

Bewertung: 1,0

3.3.4.3 Funktionsweise bei nicht-verfügbarem JavaScript

Das Symbolleisten-Steuerelement funktioniert unabhängig von der Verfügbarkeit von JavaScript.

Bewertung: 1,0

3.3.4.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Für Symbolleistenschaltflächen können Symbole eingesetzt werden, wobei es unterstützt wird, alternative Texte und Tooltips dafür zu spezifizieren. Zudem werden keine Image-Maps oder Multimedia-Präsentationen verwendet.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponenten im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Es bestehen umfangreiche Möglichkeiten um globale oder auch symbolleistenelementlokale CSS-Vorgaben anzuwenden, da sämtliche CSS-Attribute sowohl steuerelementglobal, als auch symbolleistenelementlokal festgelegt werden können. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung dieser Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von dieser Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

ToolbarControl-Komponenten basieren standardmäßig auf einem div-Element, in welchem die einzelnen Symbolleistenelemente der Reihe nach angeordnet sind, sodass keinerlei Zweckentfremdung von Tabellen vorliegt.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS und/oder von JavaScript ist die Anwendung weiterhin ohne Einschränkungen verwendbar.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext des evaluierten Steuerelements nicht anwendbar, da keine zeitgesteuerten Funktionalitäten vorliegen.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine Ansteuerung mit der Tastatur ist sowohl über die Kombination von Tab- und Entertaste als auch über Accesskeys oder via Mausklick möglich.

Bewertung: 1,0

Anforderung 10

Diese Komponenten setzen keine Popups ein.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die Symbolleisten-Steuerelemente selbst machen von keinerlei Frames Gebrauch. Zur verbesserten Orientierung und Navigation werden Submit-Elemente als Symbolleistenschaltflächen verwendet, welche ein direktes Anspringen über die Tabtaste und eine Auswahl über die Entertaste ermöglichen. Dies ermöglicht es auch unabhängig von JavaScript eine Auswahl der Navigationsmöglichkeiten über die Kombination aus Tab- und Entertaste, über Accesskeys oder per Mausklick vorzunehmen.

Auch die restlichen Symbolleistenelemente wie Textfelder sind anhand der distinkten HTML-Element-Attribut-Kombinationen, z.b. input-text, eindeutig im Sinne einer Navigation erkennbar und über Maus und Tastatur ansteuerbar.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext dieser Steuerelemente nicht relevant. Besondere Meta-Informationen werden nicht bereitgestellt. Die geforderte Übersichtlichkeit der Navigationselemente wird jedoch durch die horizontale bzw. vertikale Aneinanderreihung der Symbolleistenelemente geschaffen, wobei die Differenzierbarkeit durch die Darstellung der Symbolleistenelemente als input-Elemente gegeben ist. Die einzelnen Navigationselemente lassen sich über Tab- und Entertaste, Accesskeys oder per Mausklick ansteuern. Sprungziele bei Hyperlinks sind neben der textuellen Beschreibung auch über Tooltips illustrierbar. Zusammengehörige Elemente können über Separatoren auch visuell als zusammengehörig gekennzeichnet werden.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.4.5 Browserinteroperabilität

Das jeweilige Steuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.5 Hierarchische Steuerelemente (Tree Views)

3.3.5.1 Beschreibung

Hierarchische Strukturen können über das TreeView-Steuerelement (vgl. Abbildung 6) realisiert werden. Dieses dienst primär der Anzeige von Navigationsstrukturen, kann aber auch anderweitig eingesetzt werden.

Abbildung 6: Das eGovWDF-TreeView-Steuerelement.

3.3.5.2 Rich-Client-Funktionalität

Request-Lebenszyklus-Persistenz

Der Expansionszustand des Steuerelements bleibt nach einem Submit erhalten, unabhängig davon ob JavaScript verfügbar ist. Auch ein Aktivieren und Deaktivieren von JavaScript während des Request-Lebenszyklus wird korrekt verarbeitet, ohne dass der Komponentenzustand verloren geht.

Bewertung: 1,0

Anpassbarkeit

Das TreeView-Steuerelement ist flexibel anpassbar. So können Style-Klassen global festgelegt werden, oder aber auch die Darstellung für jedes einzelne Baumknoten-Element angepasst werden. Darüber hinaus ist es möglich, einem TreeView-Node (Baumknoten) neben Text auch Symbole hinzuzufügen. Zudem wird über ein Adapterkonzept und Vererbung die Einbettung benutzerdefinierter Steuerelemente als Baumknoten unterstützt.

Bewertung: 1,0

Verhalten

Ein Anklicken eines TreeView-Knotens bewirkt abhängig von der Konfiguration eine Navigation oder die Verarbeitung eines serverseitigen Ereignisses. Unterelemente werden per Klick auf das Expansions- bzw. Kollabierungssymbol angezeigt bzw. verborgen. Sämtliche Funktionalität steht dabei unabhängig von der Verfügbarkeit von JavaScript zur Verfügung.

Bewertung: 1,0

Entwicklungsunterstützung

Der Aufbau der Baumstrukturen wird durch den sogenannten TreeViewNode-Auflistungs-Editor auf einfache Art und Weise ermöglicht, wobei durch die Nodes-Auflistung eines Knoten rekursiv weitere Unterknoten definiert werden können.

Bewertung: 1,0

3.3.5.3 Funktionsweise bei nicht-verfügbarem JavaScript

Die TreeView-Komponente stellt sämtliche Funktionalität unabhängig von der Verfügbarkeit bzw. dem Aktiviertheitszustand von JavaScript zur Verfügung.

Bewertung: 1,0

3.3.5.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Bei Spezifikation von Baumknotensymbolen können Alternativtexte spezifiziert werden.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponenten im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass für die Komponente global diverse graphische Einstellungen vorgenommen werden können. Sämtliche globalen Darstellungseinstellungen können zudem baumknotenlokal überschrieben werden. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung dieser Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von dieser Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

Es findet keine Zweckentfremdung des table-Elements bei dieser Komponente statt. Es erfolgen Ausgaben im Sinne der Barrierefreiheit über für Aufzählungen und Listen funktional dafür vorgesehene ul- und li-Konstrukte, wobei eine beliebige TreeView-Knoten-Rekursionstiefe durch eine beliebig tiefe Verschachtelung von ul- und li-Elementen unterstützt wird.

Bewertung: 1,0

Anforderung 6

Das TreeView-Steuerelement stellt sämtliche Funktionalität unabhängig von der Verfügbarkeit von JavaScript oder CSS zur Verfügung.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext des evaluierten Steuerelements nicht anwendbar, da keine zeitgesteuerten Funktionalitäten vorliegen.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine Ansteuerung mit der Tastatur wird über Accesskeys, oder die Kombination von Tab- und Entertaste unterstützt. Zudem handelt es sich bei allen Baumknoten um Submitelemente, sodass zwar clientseitige Ereignisbehandlungsroutinen spezifiziert werden können, aber bei Nicht-Verfügbarkeit die serverseitigen Entsprechungen ausgeführt werden können, wodurch die Funktionalität stets gewahrt bleibt.

Bewertung: 1,0

Anforderung 10

Diese Komponenten setzen keine Popups ein.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die TreeView-Steuerelemente selbst machen von keinerlei Frames Gebrauch. Es erfolgen Ausgaben im Sinne der Barrierefreiheit über für Aufzählungen und Listen funktional dafür vorgesehene ul- und li-Konstrukte, wobei eine beliebige TreeView-Knoten-Rekursionstiefe durch eine beliebig tiefe Verschachtelung von ul- und li-Elementen unterstützt wird. Die Navigationelemente selbst werden über a-Elemente bzw. Submit-Buttons abgebildet.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext dieser Steuerelemente nicht relevant. Es erfolgen Ausgaben im Sinne der Barrierefreiheit über für Aufzählungen und Listen funktional dafür vorgesehene ul- und

li-Konstrukte, wobei eine beliebige TreeView-Knoten-Rekursionstiefe durch eine beliebig tiefe Verschachtelung von ul- und li-Elementen unterstützt wird. Die Navigationelemente selbst werden über a-Elemente bzw. Submit-Buttons abgebildet. Damit liegt eine logische Strukturierung der Navigationsfunktionalität beim TreeView-Steuerelement vor.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.5.5 Browserinteroperabilität

Das TreeView-Steuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.6 Fortschrittsanzeige

3.3.6.1 Beschreibung

Länger andauernde Vorgänge können über Fortschrittsanzeige-Steuerelemente wie das ProgressPresenter-Steuerelement visualisiert werden. Dieses ermöglicht die Visualisierung von:

- AJAX-Requests
- rein clientseitigen, länger andauernden Vorgängen
- regulären Request

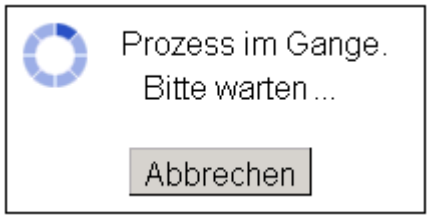


Abbildung 7: Das eGovWDF-ProgressPresenter-Steuerelement.

Falls für reguläre Requests auch eine Visualisierung im Fall der Nicht-Verfügbarkeit von JavaScript stattfinden soll, ist zusätzlich die ProgressPresenterEnforcer-Komponente einzusetzen.

Auch soll angemerkt werden, dass es unterstützt wird, AJAX-Requests oder andere clientseitige Aktivitäten über das Steuerelement und die Anzeige eines Abbrechen-Buttons oder alternativ über die Escape-Taste abzubrechen. Da bei echten Non-Ajax-Submits in den meisten Browsern ein Abbrechen lediglich über die Escape-Taste möglich ist, wird es über das ServersideProgressPresenter-Template unterstützt, einen anderen benutzerdefinierten Inhalt im Falle eines serverseitigen Requests ohne AJAX-Beteiligung anzuzeigen.

Ferner werden Komfort-Features, wie beispielsweise Sperrung der Benutzeroberfläche während laufenden Prozessen durch ProgressPresenter-Modalität unterstützt. Abbildung 7 zeigt das ProgressPresenter-Steuerelement exemplarisch.

3.3.6.2 Rich-Client-Funktionalität

Anpassbarkeit

Das ProgressPresenter-Steuerelement ist flexibel anpassbar. So kann der komplette anzuzeigende Inhalt, inklusive Animationen, völlig frei unter einem sogenannten ProgressTemplate vorgegeben werden. Da im Fall eines Non-AJAX-Reqeusts bei Nicht-Verfügbarkeit von JavaScript ein Abbrechen eines laufenden Requests über eine Schaltfläche von den Browsern nicht unterstützt wird, besteht die Möglichkeit ein anderes zu verwendendes Template für diesen Fall vorzugeben, indem beliebiger Inhalt unter dem ServersideProgressTemplate des ProgressPresenter-Steuerelements festgelegt wird.

Bewertung: 1,0

Verhalten

Bei Ausführung eines beliebigen AJAX-Requests oder eines echten Submits, falls entsprechend konfiguriert, sowie bei benutzerdefinierten clientseitigen Aktivitäten, wird das ProgressPresenter-Steuerelement eingeblendet und zeigt die benutzerdefiniert spezifizierbare Meldung und ggf. Animation an, welche den Benutzer über einen gerade laufenden Prozess informiert. Auch bei rein clientseitigen Aktivitäten kann eine Fortschrittsanzeige angezeigt werden, was aber manuell von den jeweiligen Komponenten auszulösen ist.

Bewertung: 1,0

Entwicklungsunterstützung

Der Inhalt des Steuerelements kann sowohl über die Codeansicht als auch über den Designer vorgegeben werden.

Bewertung: 1,0

3.3.6.3 Funktionsweise bei nicht-verfügbarem JavaScript

Bei deaktiviertem JavaScript kann bei entsprechender Konfiguration eine Visualisierung des rein serverseitigen Non-AJAX-Submits erfolgen. Auch ein Abbrechen des Requests wird browserabhängig über Standard-Tastenkombinationen wie Escape unterstützt.

Bewertung: 1,0

3.3.6.4 Barrierefreiheit nach BITV

Anforderung 1

Keine der Komponenten gibt Audio- oder Videoinformationen aus. Der bei laufenden Prozessen anzuzeigende Inhalt liegt damit vollständig in der Hand des Entwicklers. Das Container-Gerüst für die Anzeige selbst verwendet standardmäßig keine Audio-, Video- oder Bildinformationen.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponenten im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Die Wahl der Farben verbleibt somit in der Verantwortung des Entwicklers. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass aufgrund der freien Anpassbarkeit der anzuzeigenden Inhalte durch den Entwickler, keine Bewertungsgrundlage hinsichtlich dieses Aspekts vorliegt, sodass wie beim vorherigen Punkt die maximale Leistungsstufe ohne Abzug vergeben wird. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung dieser Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von dieser Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

Die anzuzeigenden Inhalte werden komplett durch den Entwickler vorgegeben. Lediglich der Rahmen, wobei es sich hier um einen Nicht-Tabellenbereich, genauer um einen div-Bereich, handelt, wird durch die Komponente vorgegeben.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS ist die Anwendung weiterhin ohne Einschränkung benutzbar. Eine Deaktivierung von JavaScript ist bei entsprechender Konfiguration ebenfalls ohne Einfluss, da auch Nicht-AJAX-Requests visualisiert werden können. Hierzu ist jedoch auch die ProgressPresenterEnforcer-Komponente einzusetzen.

Bewertung: 1,0

Anforderung 7

Seiteneffekte bei zeitgesteuerten Aktualisierungen wie z.B. Bildschirmflimmern treten bei Verwendung dieser Komponente nicht auf. Zudem werden zeitliche, andauernde Prozesse durch diese Komponente visualisiert. Auch Screenreader-Benachrichtigungen werden unterstützt.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt. Durch Screenreader-Notifications wird jedoch eine zusätzliche Verbesserung der Barrierefreiheit

erreicht, da damit Screenreader und deren Nutzer über aus Client-Scripting-resultierenden Bildschirmaktualisierungen informiert werden.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine Ansteuerung mit der Tastatur wird unterstützt, sodass bei AJAX-Requests ein Abbrechen über die die Fokussierung und Betätigung der Abbrechen-Schaltfläche oder ein Drücken der Escape-Taste ermöglicht wird. Bei serverseitigen Requests wird optional ein Abbrechen über die Escape-Taste browserabhängig unterstützt.

Bewertung: 1,0

Anforderung 10

Die Anzeige der Fortschrittsanzeige erfolgt als In-Page-Einblendung, d.h. es wird, auch wenn der optische Eindruck erweckt wird, kein Popup geöffnet.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Es werden keine Frames eingesetzt. Die Strukturierung von Daten des Steuerelements obliegt aufgrund der freien Anpassbarkeit des Entwicklers.

Bewertung: 1,0

Anforderung 13

Anforderung 13 ist im Kontext dieser Steuerelemente nicht relevant. Die Fortschrittsvisualisierung kann zusätzlich in Form einer Screenreader-Notification bekannt gemacht werden. Der Status (Prozess im Gange bzw. kein Prozess aktuell laufend) kann zudem programmatisch (client- und serverseitig) ermittelt werden.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.6.5 Browserinteroperabilität

Das jeweilige Steuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

3.3.7 Infotips

3.3.7.1 Beschreibung

Zur Realisierung eines Infotips soll auf die eGovWDF-InfoTip-Komponente (vgl. Abbildung 8) näher eingegangen werden. Diese erlaubt es für eine beliebige Menge von Komponenten einen beliebigen Hilfeinhalt, z.B. in Form eines Hilfetextes und eines Hilfeicons, anzuzeigen. Sowohl die um die Hilfeanzeige zu erweiternden Komponenten als auch die Hilfeanzeige selbst können über beliebige Komponenten realisiert werden, die dem TriggerTemplate bzw. dem BodyTemplate der InfoTip-Komponente untergeordnet werden. Das dynamische Ein- und Ausblenden der Hilfeinformation erfolgt bei Bewegung der Maus über das jeweilige Element und wurde über CSS realisiert, sodass keinerlei JavaScript oder serverseitige Logik zur Ausführung der Anzeigefunktionalität erforderlich ist.

Abbildung 8: Das eGovWDF-InfoTip-Steuerelement.

3.3.7.2 Rich-Client-Funktionalität

Anpassbarkeit

Sowohl die um die Hilfeanzeige zu erweiternden Komponenten als auch die Hilfeanzeige selbst können über beliebige Komponenten realisiert werden, die dem TriggerTemplate bzw. dem BodyTemplate der InfoTip-Komponente untergeordnet werden. Damit können beliebige Darstellungen realisiert werden.

Bewertung: 1,0

Verhalten

Bei Bewegung des Mauszeigers auf ein entsprechendes im TriggerTemplate des InfoTip festgelegtes Element wird unabhängig von der JavaScript-Verfügbarkeit auf Clientseite der im BodyTemplate des InfoTip festgelegte Inhalt eingeblendet und beim Wegbewegen des Mauszeigers von den im Trigger'Template fesgelegten Komponenten ausgeblendet.

Bewertung: 1,0

Entwicklungsunterstützung

Es wird eine komfortable Entwicklungsunterstützung innerhalb Visual Studio durch Vorlageneditoren ermöglicht. Damit wird es im über SmartTags aktivierbaren Vorlagenbearbeitungsmodus ermöglicht, per Drag & Drop die jeweiligen Templates zu befüllen und damit die die InfoTip-Anzeige auslösenden Komponenten als auch die anzuzeigenden Komponenten graphisch festzulegen.

Bewertung: 1,0

3.3.7.3 Funktionsweise bei nicht-verfügbarem JavaScript

Die Funktionalität des InfoTips ist unabhängig von der Verfügbarkeit von JavaScript gegeben.

Bewertung: 1,0

3.3.7.4 Barrierefreiheit nach BITV

Anforderung 1

Die Komponente gibt keine Audio- oder Videoinformationen aus. Zudem werden keine Image-Maps oder Multimedia-Präsentationen verwendet. Die Spezifikation von Bildern liegt aufgrund der freien Befüllung des InfoTip-Steuerelement-Inhaltsbereichs in der Verantwortung des Entwicklers und damit außerhalb der Komponente selbst.

Bewertung: 1,0

Anforderung 2

Da keine spezifische Darstellung zwingend vorgegeben wird, kann auch keinerlei negative Bewertung der Komponenten im Hinblick auf die Verständlichkeit bei Elimination des Aspekts Farbe vorgenommen werden. Über weitere Komponenten des eGovWDF-Frameworks wie das ThemingControl und die ASP.NET-Theming-Mechanismen können jedoch z.B. explizit kontrastreiche Ansichten der Anwendung bereitgestellt werden, wobei ein Wechsel zwischen Normalversion und Kontrastversion einer Anwendung zur Laufzeit unterstützt wird.

Bewertung: 1,0

Anforderung 3

Zur Anforderung 3 ist zu bemerken, dass aufgrund der freien Anpassbarkeit der anzuzeigenden Inhalte durch den Entwickler, keine Bewertungsgrundlage hinsichtlich dieses Aspekts vorliegt, sodass wie beim vorherigen Punkt die maximale Leistungsstufe ohne Abzug vergeben wird. Das Komponentengerüst entspricht den aktuellen Standards wie XHTML und CSS.

Bewertung: 1,0

Anforderung 4

Mehrsprachigkeit wird nicht explizit unterstützt, sodass auch für diesen Punkt kein Anwendungsbereich im Zusammenhang mit der Evaluierung dieser Komponente vorliegt. Die Festlegung von Akronymen und Abkürzungen liegt in der Zuständigkeit des Entwicklers und ist unabhängig von dieser Komponente zu betrachten.

Bewertung: 1,0

Anforderung 5

Der Komponenteninhalt kann völlig frei und ohne Tabellenelemente festgelegt werden, wodurch sich Zweckentfremdungen (von Tabellenstrukturen) im Rahmen der Realisierung der Komponente ausschließen lassen.

Bewertung: 1,0

Anforderung 6

Bei Deaktivierung von CSS besteht die Funktionalität zum dynamischen Ein- und Ausblenden der Hilfeinformationen nicht mehr, da die Hilfeinformationen dann ständig, in der Regel rechts neben der Komponente, eingeblendet werden. Dies ist jedoch als kein Nachteil im Sinne der Barrierefreiheit zu bewerten, da der Zusammenhang zwischen Hilfeinformation und zugehörigen die Hilfe auslösenden Komponenten aufgrund der räumlichen Anordnung weiterhin erkennbar ist und über Header-Zeilen mit Komponentenbezug in der InfoTip-Komponente verdeutlicht werden kann. Zudem stehen die Hilfeinformationen dann ständig zur Verfügung. Die Verfügbarkeit von JavaScript hat keinerlei Einfluss auf die Funktionsweise der Komponente.

Bewertung: 1,0

Anforderung 7

Diese Anforderung ist im Kontext des evaluierten Steuerelements nicht sinnvoll anwendbar, da keine zeitgesteuerten Funktionalitäten vorliegen. Effekte wie Bildschirmflackern treten nicht auf.

Bewertung: 1,0

Anforderung 8

Für Aspekt 8 gilt analog zu Aspekt 7, dass keine spezifische Anwendbarkeit vorliegt.

Bewertung: 1,0

Anforderung 9

Es werden keine gerätespezifischen Eventhandler oder anderweitigen Funktionen benutzt, die einen Einsatz des Frameworks auf ein bestimmtes Ein- oder Ausgabegerät beschränken. Eine browserübergreifende Ansteuerung mit der Tastatur ist möglich, wenn eine entsprechende Konfiguration in den CSS-Einstellungen dahingehend vorgenommen wird, dass für die Anzeige statt der hover-Pseudoklasse die focus- und die active-Pseudoklasse verwendet wird.

Bewertung: 1,0

Anforderung 10

Diese Komponenten setzen keine echten Popups ein.

Bewertung: 1,0

Anforderung 11

Die erzeugte (X)HTML-Ausgabe der Komponente basiert auf öffentlichen, zugänglichen und vollständig dokumentierten Webtechnologien, wie XHTML und CSS.

Bewertung: 1,0

Anforderung 12

Die InfoTip-Steuerelemente selbst machen von keinerlei Frames Gebrauch, wenngleich diese Möglichkeit optional besteht, um fremde Inhalte einzubinden. Die Einbringung von Elementen zur Strukturierung bzw. Verbesserung der Navigation wird unterstützt.

Bewertung: 1,0

Anforderung 13

Die Komponente erlaubt die Spezifikation beliebiger Meta-Informationen. Eine sinnvolle Gruppierung von zusammengehörigen (Navigations-Elementen), wie Hyperlinks wird von der Komponente unterstützt.

Bewertung: 1,0

Anforderung 14

Das Sprachniveau und die Darstellung der Texte liegen in der Hand des jeweiligen Entwicklers.

Bewertung: 1,0

3.3.7.5 Browserinteroperabilität

Das jeweilige Steuerelement funktioniert unter allen spezifizierten Browsern.

Internet Explorer 6.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Mozilla Firefox 2.0+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Opera 9+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

Safari 3.1+

Es liegen keine funktionalen Einschränkungen vor.

Bewertung: 1,0

4 Auswertung

4.1 Beschreibung

In diesem Abschnitt soll auf Basis der gewonnen Daten ein Vergleich zwischen dem eGovWDF-Rich Client-Komponenten-Framework und den in Kern (2008a) untersuchten Frameworks erfolgen. Zunächst wird eine oberflächliche Analyse auf Basis der erreichten Gesamtanforderungserfüllung vorgenommen (Abschnitt 2 bzw. Anlagen A1 bis A5). Im Anschluss daran wird eine komponentenzentrierte Betrachtung der Gesamtanforderungserfüllung vorgenommen (Anlagen B1 bis B3). Im nächsten Schritt wird eine anforderungsartbezogene (bereichsbezogene) Sichtung der Ergebnisse durchgeführt (Anlagen C1 bis C4). Ferner wird in Anlage D eine Tabelle der Detailergebniswerte angeführt, welche jedoch vordergründig nicht dem Primärziel, der Bestätigung der in Kapitel 2 formulierten These, dient, sondern eine Datenbasis für zukünftige Analysen sowie eine Orientierung für eine bedarfsgesteuerte Erforschung der zugrundeliegenden Themenbereiche liefern soll.

4.2 Überblicks-Analyse

In Kern (2008a) wurden ein mittlerer Gesamtanforderungserfüllungsgrad von 47% und ein maximaler Gesamtanforderungserfüllungsgrad von 64% (beim DevExpress ASPxperience-Framework) festgestellt. Der maximal erreichte Grad der JS-Unabhängigkeit liegt bei Berücksichtigung aller getesteten Frameworks (ausschließlich eGovWDF) bei 14%, was bedeutet, dass beim Framework mit dem höchsten Grad an Anforderungserfüllung 14% aller im Rahmen der Evaluierung geprüften Komponenten unabhängig von der Verfügbarkeit von JavaScript ihre Funktion wahren. Berechnet man den Anforderungserfüllungsgrad auf Basis aller Komponenten aller Frameworks, wird ein Wert von 4% erreicht, was bedeutet, dass 4% aller geprüften Komponenten aller geprüften Frameworks unabhängig von JavaScript funktionieren. Besonders dieser sehr niedrige Wert der JS-Unabhängigkeit stellt ein Ausschlusskriterium für den Einsatz der geprüften Frameworks dar, weil damit eine zentrale Anforderung der Barrierefreiheit verletzt wird. Im Vergleich dazu erreicht das Rich Client-Komponententeilframework von eGovWDF einen Gesamtanforderungserfüllungsgrad von 100%, eine JS-Unabhängigkeit von 100% sowie eine Komponentenabdeckung von 100%. Hierbei ist darauf hinzuweisen, dass eGovWDF explizit mit der Intention entwickelt worden ist, die in Kern (2008a) formulierten Anforderungen zu erfüllen, sodass während der Entwicklung in mehreren Iterationszyklen eine stetige Erhöhung der

genannten Werte erreicht werden konnte, bis schließlich 100% bei allen Werten realisiert wurde. Abbildung 9 bzw. Anlage A2 veranschaulichen die genannten Ergebniswerte der Untersuchung und stellen diese den Ergebniswerten von eGovWDF gegenüber, wobei auf der Abszissenachse die Frameworks und auf der Ordinatenachse die erreichten Werte der Gesamtanforderungserfüllung, JS-Unabhängigkeit bzw. Komponentenabdeckung aufgetragen sind.

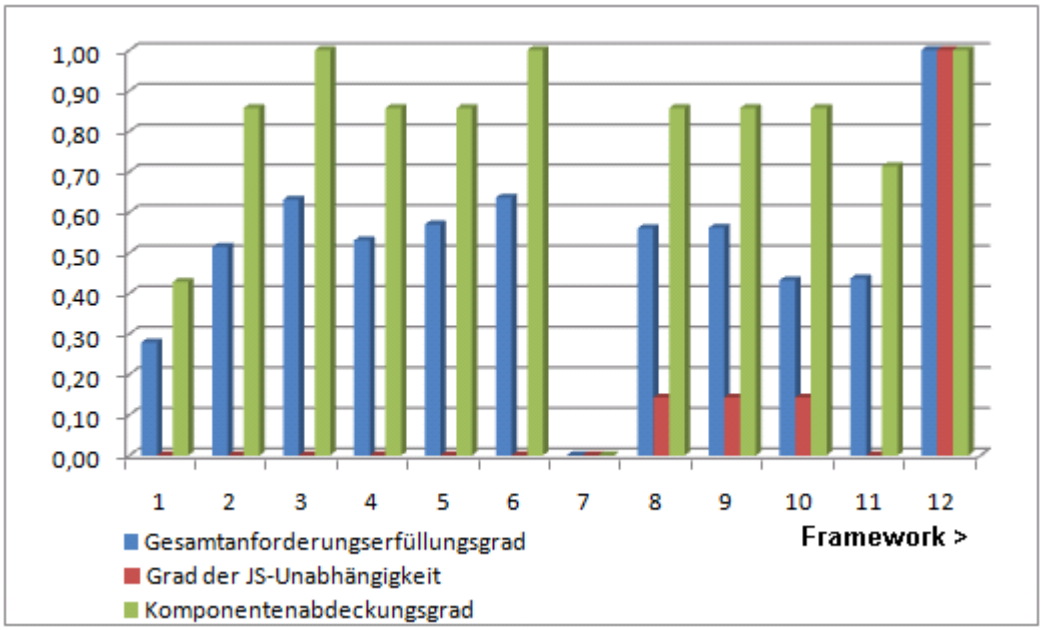


Abbildung 9: Vergleich der Erfüllungsgrade aktueller Frameworks und eGovWDF.

Legende:

- Framework 1: ASP.NET 3.5 (Standard)
- Framework 2: AJAX Control Toolkit
- Framework 3: ComponentArt Web.UI
- Framework 4: NetAdvantage for ASP.NET
- Framework 5: Telerik Controls for ASP.NET AJAX
- Framework 6: DevExpress ASPxpericence
- Framework 7: .JSF (Standard)
- Framework 8: MyFaces Tomahawk
- Framework 9: JBoss RichFaces
- Framework 10: Oracle ADF Faces Rich Client
- Framework 11: NetAdvantage for JSF
- Framework 12: eGovWDF

Abbildung 10 bzw. Anlage A3 verfeinern diese Darstellung, indem darin weitere statistische Kennzahlen zur sich aus der Betrachtung der verschiedenen Frameworks ergebenden Verteilung, visualisiert werden. Dabei werden zwei Boxen samt Whisker in einem Boxplot dargestellt, wobei die erste Darstellung die Verteilung der Gesamtanforderungserfüllung auf alle Frameworks, aber ohne eGovWDF, zeigt. Die zweite Grafik hingegen beinhaltet auch die Gesamtanforderungs-erfüllung von eGovWDF als Wert der Verteilung.

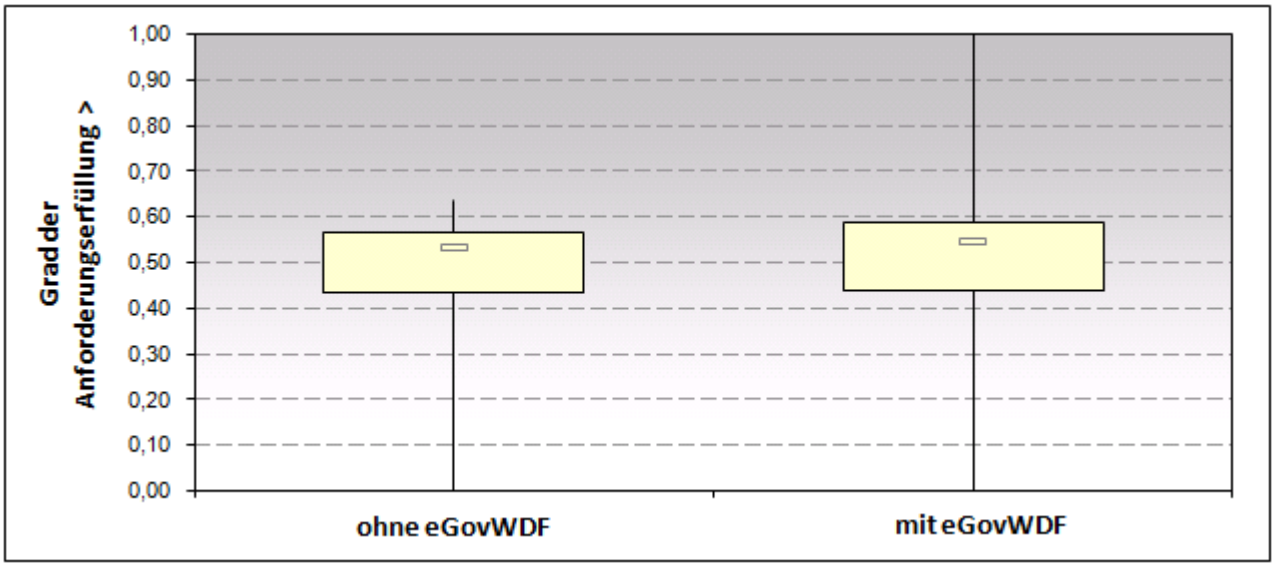


Abbildung 10: Boxplot-Visualisierung der Gesamtanforderungserfüllung mit/ohne eGovWDF.

Die linke Box samt Whisker zeigt einen nur geringen Abstand von oberem Quartil, unterem Quartil und Median. Zudem verdeutlicht der nach oben zeigende Whisker, dass zwar Ausreiser nach oben vorliegen, diese aber nur unwesentlich vom Median abweichen, welcher eine mittlere Gesamtanforderungserfüllung in Höhe von 0,53 (53%) besagt. Daraus kann der Schluss gezogen werden, dass sich alle der aktuell verfügbaren, untersuchten Frameworks - mit Ausnahme einiger durch den nach unten zeigenden Whisker repräsentierten wesentlich schlechter abschneidenden Frameworks - auf dem gleichen Niveau, dem Stand der Technik, bewegen und eine herausstechende, die besonderen Anforderungen im Bereich des eGovernment erfüllende Lösung, nicht vorliegt.

Bei Betrachtung der zweiten Box samt Whisker ist der Gesamtanforderungserfüllungsgrad von eGovWDF berücksichtigt. Wie aus der Darstellung erkennbar ist, sind Median, oberes und unteres Quartil leicht erhöht. Diese Ähnlichkeit zur ersten Darstellung verdeutlicht die Störunanfälligkeit der genannten statistischen Kennzahlen. Ausschlaggebend in dieser Darstellung ist jedoch, dass der obere Whisker bis zum Maximalwert der Gesamtanforderungserfüllung, also 1,00 bzw. 100%,

reicht. Der dabei visualisierte Extremwert des Whisker von 1,0 repräsentiert dabei die Gesamtanforderungserfüllung von eGovWDF und verdeutlicht damit, dass eGovWDF die oben angesprochene, fehlende, herausstechende und die besonderen Anforderungen im Bereich des eGovernment erfüllende Lösung über dem Stand der Technik, darstellt.

Ein ähnliches, aber wesentlich drastischeres Bild zeigt sich bei analoger Betrachtung des Grads der JS-Unabhängigkeit – einmal ohne und einmal mit Berücksichtigung von eGovWDF. Wie in Abbildung 11 bzw. Anlage A4 dargestellt, besitzen sowohl unteres Quartil, Median als auch unterer Whisker den Wert 0, was eine Indikation für das komplette Fehlen von JavaScript-Unabhängigkeit bei nahezu allen Komponenten aller getesteter Frameworks ist. Auch reicht der obere Whisker in der ersten Abbildung lediglich bis 14%, was bedeutet, dass selbst beim Framework mit dem größten Grad an JS-Unabhängigkeit lediglich ein Wert von 14% erreicht wird. Bei Betrachtung der zweiten Teilabbildung von Abbildung 11 ergibt sich ein ähnliches Bild, wobei durch die Berücksichtigung von eGovWDF das obere Quartil höher als bei der ersten Teildarstellung ist und der obere Whisker bis zu 1,00 und damit 100% reicht. Der Wert von 100% repräsentiert eGovWDF und verdeutlicht bei komparativer Betrachtung mit der eGovWDF nicht berücksichtigenden Darstellung, dass eGovWDF sich hinsichtlich des Aspekts der JS-Unabhängigkeit wesentlich über dem Stand der Technik befindet.

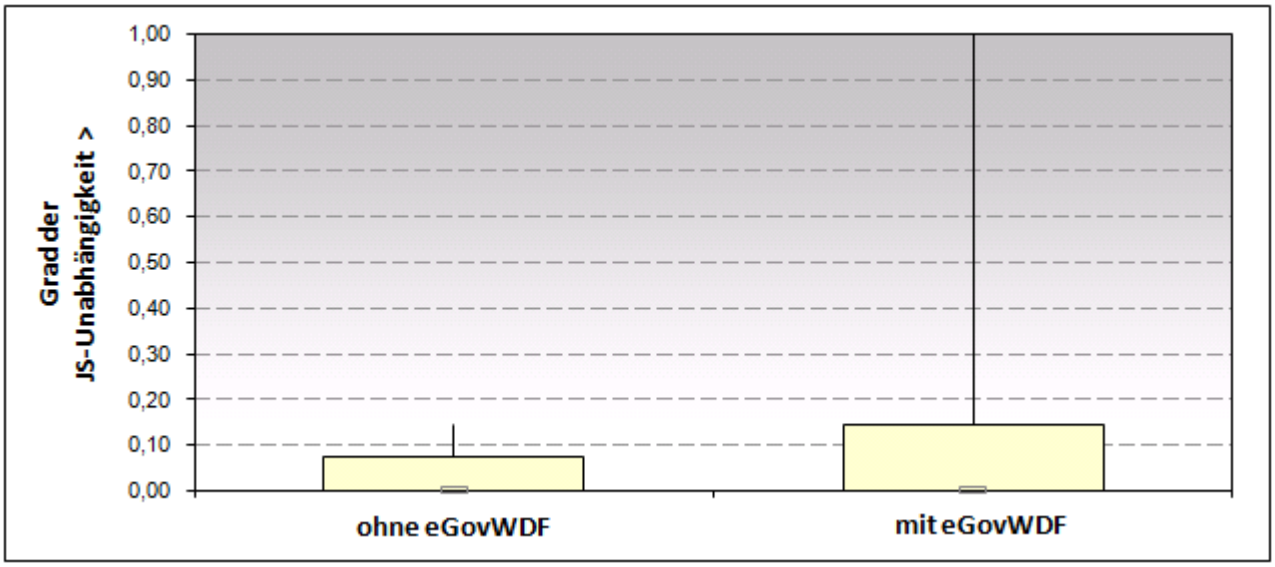


Abbildung 11: Boxplot-Visualisierung der JS-Unabhängigkeit mit/ohne eGovWDF.

Ein wesentlich anderes Bild ergibt sich bei Analyse des Komponentenabdeckungsgrad der in Kern (2008a) untersuchten Frameworks. Wie in Abbildung 12 bzw. Anlage A5 visualisiert wird, liegt ein im Mittel sehr hoher Grad an Komponentenabdeckung vor, was bedeutet, dass die geprüften

Frameworks die im Rahmen der Studie geforderten Komponenten größtenteils beinhalten. Anhand des oberen Whiskers kann zudem festgestellt werden, dass unabhängig von eGovWDF eines oder mehrere Frameworks die maximale Komponentenabdeckung besitzen. Dies bestätigt die Güte des Frameworkauswahlprozesses der Studie und ist ein Indikator dafür, dass tatsächlich Frameworks untersucht worden sind, die im Hinblick auf das Anforderungsprofil auch relevant sind und das Potential einer größtmöglichen Anforderungsbefriedigung besitzen.

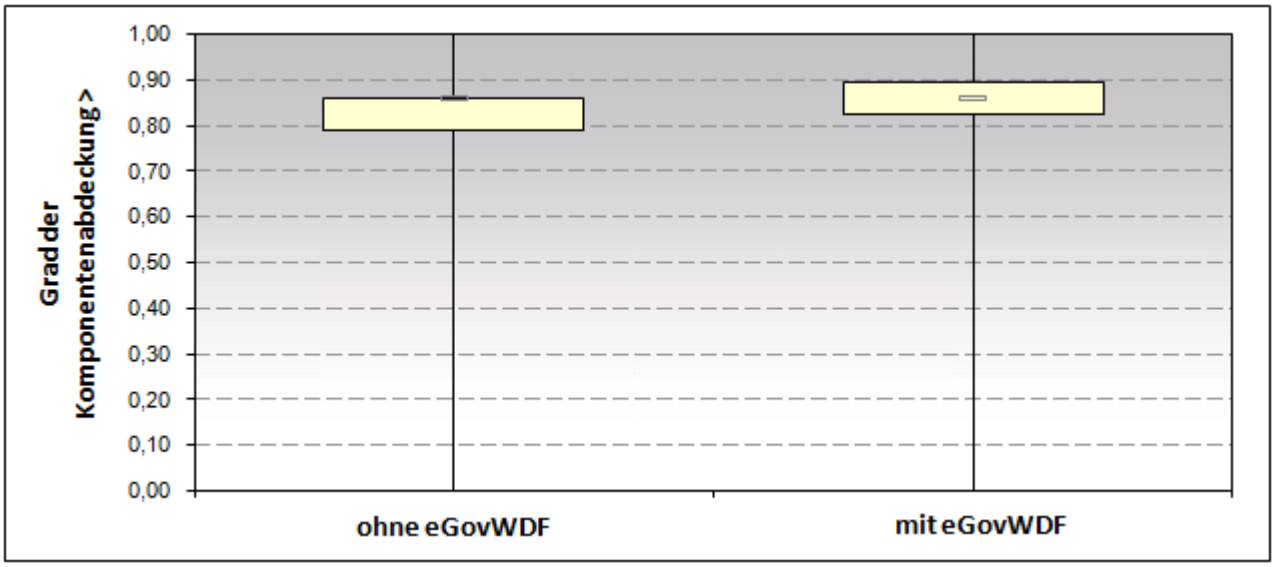


Abbildung 12: Boxplot-Visualisierung des Komponentenabdeckungsgrads mit/ohne eGovWDF.

4.3 Anforderungsartbezogene Analyse

In diesem Bereich soll untersucht werden, in welchen Anforderungsbereichen (Anforderungsarten) die Primär-Unterschiede zwischen der Anforderungserfüllung von eGovWDF und den aktuellen Lösungen vorliegen. Dazu werden pro Framework und Anforderungsbereich der Erfüllungsgrad kummulativ über alle Komponenten ermittelt und die errechneten Werte einander gegenüber gestellt.

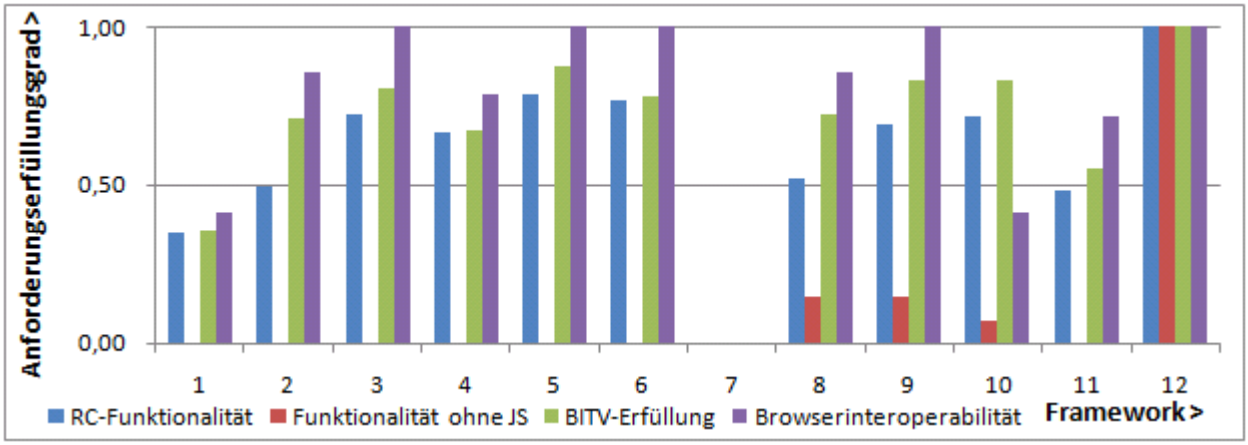


Abbildung 13: Anforderungserfüllung bezogen auf Frameworks und Anforderungsarten.

Wie durch Abbildung 13 und Anlage C2 angedeutet wird, wird von den Frameworks 1 bis 11 (Framework 12 stellt eGovWDF dar) eine relativ hohe Anforderungserfüllung (Mittelwert: 73%) im Bereich der Browserinteroperabilität erreicht, wobei die geprüften Browser Internet Explorer 6.0+, Firefox 2.0+, Opera 9 und Safari 3.1 umfassen. Im Bereich Rich Client-Funktionalität wird hingegen im Durchschnitt eine Anforderungserfüllung von 56% erreicht. Betrachtet man den Themenbereich BITV ist ein Durchschnittswert von 65% festzuhalten. Eine sehr geringe Anforderungserfüllung liegt beim Aspekt JavaScript-Unabhängigkeit vor, da im Mittel lediglich 3% der Komponenten eines Frameworks ohne JavaScript funktionieren. Selbst beim Framework mit dem höchsten Grad an JavaScript-Erfüllung wird lediglich ein Wert von 14% erreicht.

Wenngleich der BITV-Anforderungserfüllungswert von 65% als relativ hoch bewertet werden kann, sagt dieser alleine nichts über die Erfüllung der BITV-Einzelanforderungen aus. Wie in Abbildung 14 und Anlage C4 dargestellt, resultiert der hohe Anforderungserfüllungsgrad vor allem aus den hohen Erfüllungswerten bestimmter BITV-Anforderungen. Einige andere, wie Anforderung 6 (Funktionieren der Webanwendung unabhängig davon, ob der User Agent neuere Technologien unterstützt, z.B. JavaScript) sind jedoch erheblich niedriger, was ein essentielles Problem darstellt, weil ein Totalausfall der Befriedigung einer Anforderung als Gesamtnichtkonfirmität mit der BITV bewertet werden kann. Besonders im Fall JavaScript-Unabhängigkeit ist dieses Problem deutlich in der genannten Abbildung zu erkennen, weil sowohl Median, unteres und oberes Quartil sowie unterer und oberer Whisker lediglich Minimalwerte kleiner 15% erreichen. Dies bedeutet, dass bei allen untersuchten Frameworks eine sehr niedrige Anforderungserfüllung vorliegt, sodass auch eine Kombination verschiedener Frameworks zur Erhöhung des Anforderungserfüllungsgrads nur geringe Erfolgsaussichten besitzt.

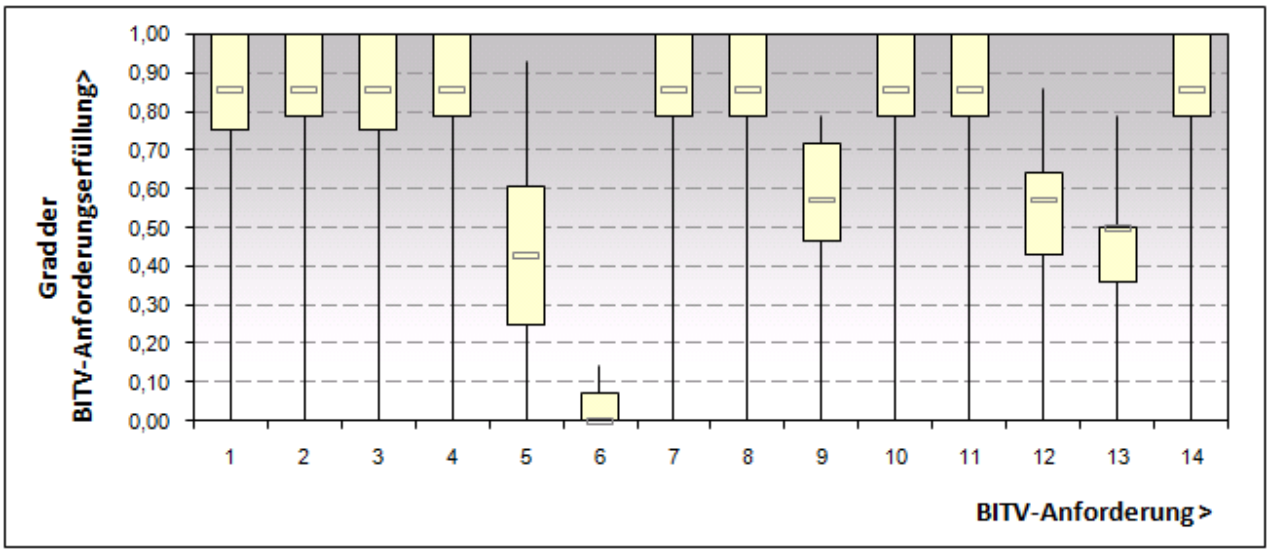


Abbildung 14: Boxplot-Visualisierung der BITV-Anforderungserfüllung.

Allgemein ist festzuhalten, dass auch wenn manche Frameworks in manchen Aspekten Ausreißer nach oben im Hinblick auf die Anforderungserfüllung besitzen, eine Kombination der entsprechenden Anforderungen nicht immer möglich ist, da die verschiedenen Frameworks und die zugehörigen Komponenten auf unterschiedlichen Technologien, Bibliotheken und Paradigmen basieren und somit falls überhaupt möglich, enormer Entwicklungsaufwand daraus resultieren kann bzw. ineffiziente Implementierungen resultierend aus der Kombination nur beschränkt kompatibler Technologien entstehen können.

Neben dieser aggregierten Betrachtung der Anforderungserfüllung über alle geprüften Komponenten eines Frameworks soll im Folgenden eine komponentenzentrierte Analyse stattfinden.

4.4 Komponentenzentrierte Analyse

In diesem Bereich soll untersucht werden, bei welchen Komponenten die Primär-Unterschiede zwischen der Anforderungserfüllung von eGovWDF und den aktuellen Lösungen vorliegen. Dazu werden pro Framework und Komponente der Erfüllungsgrad kummulativ über alle Anforderungen ermittelt und die errechneten Werte einander gegenüber gestellt.

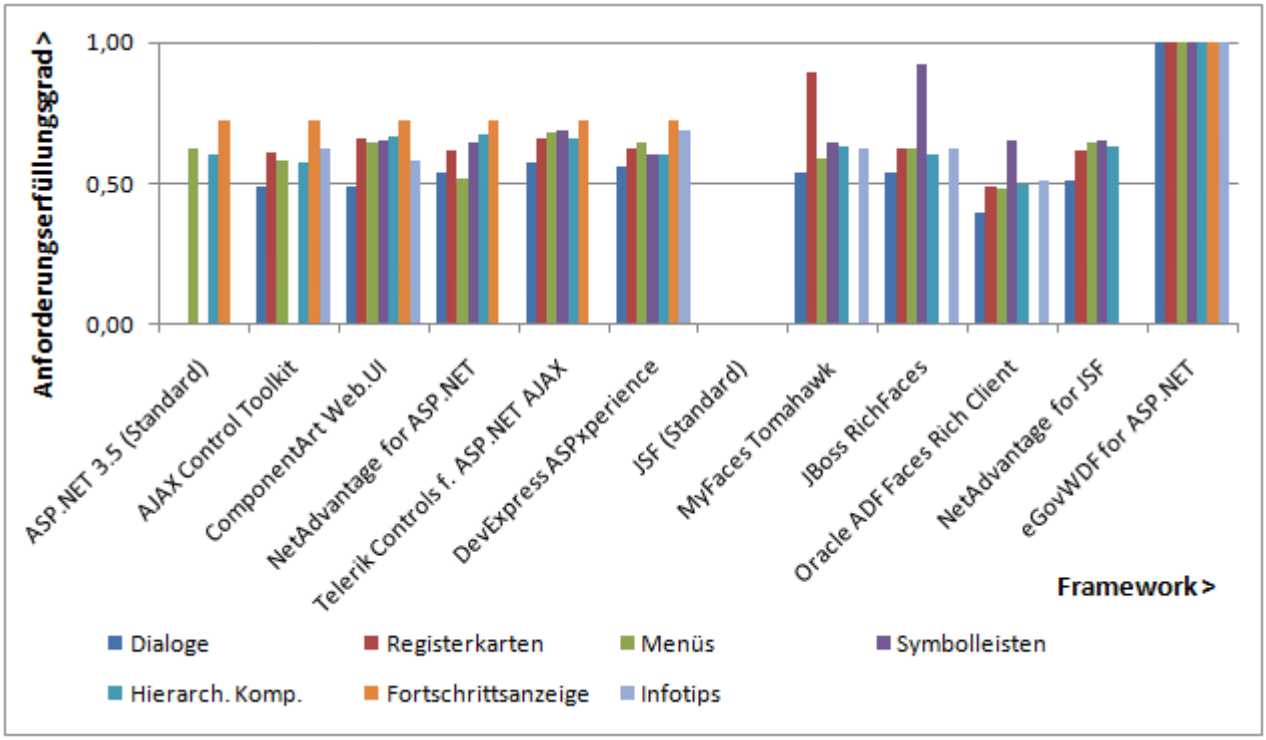


Abbildung 15: Anforderungserfüllung pro Framework mit Komponenten als Reihen.

Abbildung 15 bzw. Anlage B1 zeigen die Anforderungserfüllung bei den verschiedenen Frameworks pro Komponente an. Dabei ist erkennbar, dass ingesamt eine mittelmäßige Anforderungserfüllung festzustellen ist. Hierbei ist auch ersichtlich, dass JSF ohne weitere Komponentenbibliotheken nicht zur Realisierung reichhaltiger Webanwendungen eingesetzt werden kann. Alleine bei den Registerkarten- und Symbolleisten-Komponenten liegt eine überdurchschnittliche Anforderungserfüllung (> 75%) bei den Frameworks MyFaces Tomahawk bzw. JBoss Rich Faces vor.

In Abbildung 16 bzw. Anlage B2 werden die Anforderungserfüllungswerte aller Frameworks nach Komponenten gruppiert dargestellt. Dabei ist klar ersichtlich, dass der Anforderungserfüllungsgrad-Unterschied zwischen den aktuellen Lösungen und eGovWDF sich auf keine bestimmte Komponente beschränkt, da bei keiner Komponente über alle Frameworks hinweg ein Anforderungserfüllungsgrad nahe 100% erreicht wird, was dem Ergebnis von eGovWDF entsprechend würde.

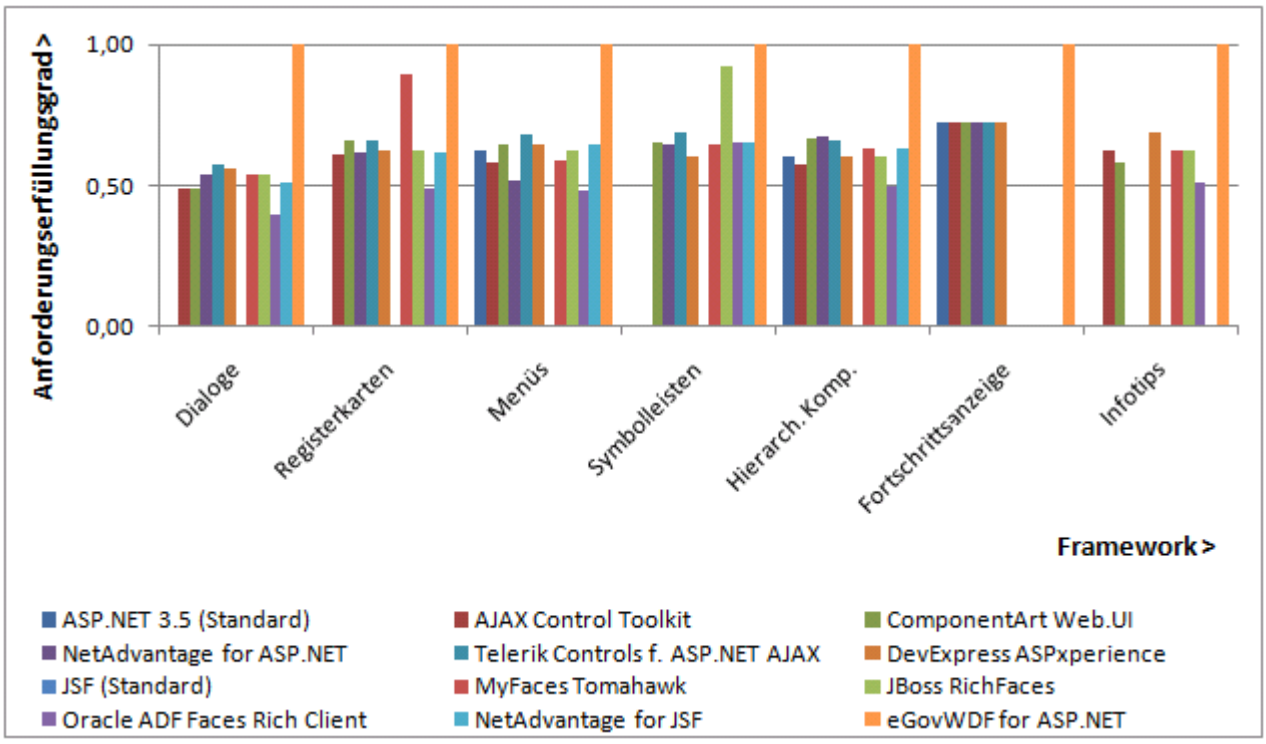


Abbildung 16: Anforderungserfüllung pro Komponente mit Frameworks als Reihen.

Weitere Details können Abbildung 17 bzw. Anlage B3 entnommen werden, wo die statistischen Werte Median, unteres Quartil, oberes Quartil, oberer und unterer Whisker anhand eines Boxplot visualisiert werden. Dieser Boxplot zeigt auch, dass der Anforderungserfüllungsgrad-Unterschied bei Dialogen und InfoTips im Mittel am größten ist, was durch Betrachtung der Komponenten-Mediane in der genannten Abbildung feststellbar ist. Ferner gibt diese, eGovWDF nicht berücksichtigende, Darstellung einen Überblick darüber, bei welchen Komponenten Ausreißer nach oben (oberer Whisker) vorliegen und damit bei welchen Komponenten zumindest in einem der untersuchten Frameworks ein im Vergleich zu den restlichen Frameworks herausragender Anforderungserfüllungsgrad vorliegt. Dies kann alleine bei Komponente 2 und Komponente 4 festgestellt werden, wobei es sich um das weiter oben schon angesprochene Registerkartensteuerelement und das Symbolleistensteuerelement handelt. Bei den restlichen Komponenten ist festzustellen, dass der obere Whisker nur unwesentlich über dem oberen Quartil liegt, sodass ohne Betrachtung von eGovWDF keine über dem Stand der Technik liegende, herausragende Lösung durch eines der untersuchten Frameworks bereitgestellt wird.

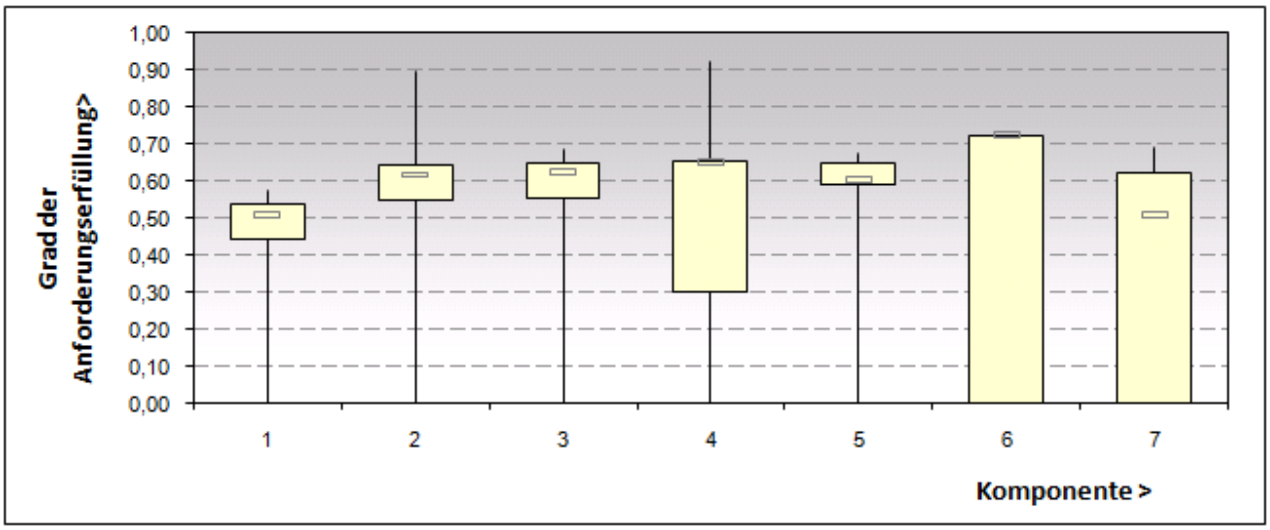


Abbildung 17: Boxplot-Visualisierung der Anforderungserfüllung pro Komponente.

Legende:

- Komponente(ntyp) 1: Dialoge
- Komponente(ntyp) 2: Registerkartensteuerelemente
- Komponente(ntyp) 3: Menüs
- Komponente(ntyp) 4: Symbolleisten
- Komponente(ntyp) 5: Hierarchische Komponenten
- Komponente(ntyp) 6: Fortschrittsanzeige
- Komponente(ntyp) 7: Infotip-Komponenten

5 Fazit

Zusammengefasst kann festgehalten werden, dass sowohl bei Betrachtung der Gesamtanforderungserfüllung als auch der Anforderungserfüllungswerte auf Anforderungsart- und Komponentenbasis keines der in Kern (2008a) untersuchten Frameworks dazu in der Lage ist, die besonderen Anforderungen im Bereich des eGovernment zu erfüllen. Auch zeigen die Visualisierungen in Kapitel 4, dass keines der in Kern (2008a) untersuchten Frameworks sich signifikant vom Stand der Technik abhebt. eGovWDF erreicht jedoch einen sehr hohen Prozentsatz an Gesamtanforderungsbefriedigung, was daraus resultiert, dass eGovWDF auf Basis der in Kern (2008a) beschriebenen Anforderungen entwickelt worden ist. Die, die in Abschnitt 2.1 gestellte Hypothese bestätigende Gesamtanforderungsbefriedigung von eGovWDF in Höhe von 100% bedeutet, dass eGovWDF sämtliche Einzelanforderungen jeder Anforderungsart und bezogen auf jede untersuchte Komponente vollständig erfüllt, wobei eGovWDF ferner als wesentlich über dem Stand der Technik bewertet werden kann, wenn die statistischen Werte des Vergleichs aller in Kern (2008a) beschriebenen Frameworks betrachtet werden (vgl. Abschnitt 4.3). Besonders große Unterschiede zwischen eGovWDF und den restlichen untersuchten Frameworks liegen im Bereich JS-Unabhängigkeit vor, wo eine maximale Anforderungserfüllung in Höhe von 14% festgestellt werden kann. Dies hat zudem einen großen Einfluss auf die Barrierefreiheit der jeweiligen Komponenten, da gemäß der BITV Webanwendungen unabhängig von der Unterstützung des User Agent des Anwenders für moderne Technologien wie JavaScript, ihre Funktion wahren müssen.

Unabhängig von dem hohen Grad der JavaScript-Unabhängigkeit ist das eGovWDF-Framework im Bereich Web Accessibility positiv hervorzuheben, da sämtliche Komponenten explizite Screenreader-Benachrichtungen und eine optimierte Textbrowser-Unterstützung vorsehen. Daneben unterstützt das Framework eine stufenlose Vergrößerung der darzustellenden Webseite und die Möglichkeit zwischen Normalansicht und kontrastreicheren Ansichten zu wechseln.

Literaturverzeichnis

[Apple Inc. 2008] Apple Inc. (Hrsg.): *Apple Human Interface Guidelines*. 2008. – URL http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/. – Letzter Zugriff am 05.05.2008

[Bayerisches Staatsministerium des Innern 2003] Bayerisches Staatsministerium des Innern (Hrsg.): *Bayerische Barrierefreie Informationstechnik-Verordnung (BayBITV)*. Juni 2003

[Benson et al. 2004] Benson, Calum ; Elman, Adam ; Nickell, Seth ; Robertson, Colin Z.: *GNOME Human Interface Guidelines 2.0*. 2004. – URL http://developer.gnome.org/projects/gup/hig/2.0/hig-2.0.pdf. – Letzter Zugriff am 18.02.2008

[Boehm 2006] Boehm, Barry: A view of 20th and 21st century software engineering. In: *ICSE '06: Proceedings of the 28th international conference on Software engineering*. New York, NY, USA: ACM, 2006, S. 12–29

[Bundesministerium des Innern 2002] Bundesministerium des Innern (Hrsg.): *Barrierefreie Informationstechnik-Verordnung (BITV)*. Juli 2002. – URL http://bundesrecht.juris.de/bundesrecht/bitv/gesamt.pdf

[Developer Express Inc. 2008a] Developer Express Inc. (Hrsg.): *ASPxperience Suite Overview*. 2008. – URL http://www.devexpress.com/Products/NET/WebForms/ASPxperience/index.xml. – Letzter Zugriff am 24.05.2008

[Developer Express Inc. 2008b] Developer Express Inc. (Hrsg.): *ASPxperience Suite - Specifications*. 2008. – URL http://www.devexpress.com/Products/NET/WebForms/ASPxperience/info.xml. – Letzter Zugriff am 24.05.2008

[Esposito 2004] Esposito, Dino: *Introducing ASP.NET 2.0*. Redmond, WA, USA : Microsoft Press, 2004

[Esposito 2005] Esposito, Dino: *Programming Microsoft® ASP.NET 2.0 Core Reference.* Microsoft Press Corp., 2005

[Gamma et al. 1994] Gamma, Erich ; Helm, Richard ; Johnson, Ralph ; Vlissides, John: *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, Massachusetts : Addison Wesley, 1994

[Geary und Horstmann 2007] Geary, David M. ; Horstmann, Cay S.: *Core JavaServer Faces (Core)*. Prentice Hall International, 2007

[Hailpern et al. 2009] Hailpern, Joshua ; Guarino-Reid, Loretta ; Boardman, Richard ; Annam, Srinivas: Web 2.0: blind to an accessible new world. In: *WWW '09: Proceedings of the 18th international conference on World wide web*. New York, NY, USA : ACM, 2009, S. 821–830

[Infragistics 2008a] Infragistics (Hrsg.): *Infragistics Announces NetAdvantage for .NET 2008 Volume 1*. 2008. – URL http://www.infragistics.com/Awesome.aspx?id=8583. – Letzter Zugriff am 12.05.2008

[Infragistics 2008b] Infragistics (Hrsg.): *NetAdvantage for JSF*. 2008. – URL http://dl-uklo.infragistics.com/products/NetAdvantage/JSF/2008.1/help/NetAdvantage_20081_JSF_Help_PDF.zip. – Letzter Zugriff am 05.06.2008

[JBoss.org 2008a] JBoss.org (Hrsg.): *JBoss RichFaces*. 2008. – URL http://www.jboss.org/jbossrichfaces/. – Letzter Zugriff am 29.05.2008

[JBoss.org 2008b] JBoss.org (Hrsg.): *RichFaces Developer Guide*. 2008. – URL http://www.jboss.org/file-access/default/members/jbossrichfaces/freezone/docs/devguide/en/pdf/richfaces-usersguide.pdf. – Letzter Zugriff am 30.05.2008

[KBSt 2006a] KBSt (Hrsg.): *SAGA 3.0: Standards und Architekturen für E-Government-Anwendungen*. Oktober 2006. – URL http://www.kbst.bund.de/cln_012/nn_836802/SharedDocs/Anlagen-kbst/Saga/saga__3__0,templateId=raw,property=publicationFile.pdf/saga_3_0.pdf

[KBSt 2006b] KBSt (Hrsg.): *V-Modell XT 1.2.1*. 2006. – URL ftp://ftp.tu-clausthal.de/pub/institute/informatik/v-modell-xt/Releases/1.2.1/Dokumentation/V-Modell-XT-Gesamt.pdf

[KDE-Entwicklerteam o.J.] KDE-Entwicklerteam (Hrsg.): *KDE 3 Styleguide*. o.J.. – URL http://developer.kde.org/documentation/standards/kde/style/styleguide.pdf. – Letzter Zugriff am 18.02.2008

[Kern 2008a] Kern, Walter: *Evaluierung bedeutender Webapplikations-Entwicklungsframeworks im Hinblick auf die Aspekte Rich-Client-Funktionalität und Barrierefreiheit im Kontext der Anforderungen im Bereich eGovernment*. Regensburg, Universität. – URL http://www.opus-bayern.de/uni-regensburg/volltexte/2008/1040/pdf/Evaluierung_Web_2.0_Frameworks.pdf. – Letzter Zugriff am 12.07.2009

[Kern 2008b] Kern, Walter: Web 2.0 - End of Accessibility? Analysis of Most Common Problems with Web 2.0 Based Applications Regarding Web Accessibility. In: *International Journal of Public Information Systems* 02 (2008), Nr. 2, S. 131–154

[Kern 2008c] Kern, Walter: Netz ohne Schranken – Barrierefreie Webseiten mit ASP.NET realisieren. In: *dotnetpro* 07 (2008), S. 73–77

[Kern 2008d] Kern, Walter: Völlig losgelöst – Webanwendungen mit Gizmox Visual WebGui. In: *dotnetpro* 10 (2008), S. 78–82

[Linaje et al. 2007] Linaje, Marino ; Preciado, Juan C. ; Sánchez-Figueroa, Fernando: Engineering Rich Internet Application User Interfaces over Legacy Web Models. In: *IEEE Internet Computing* 11 (2007), Nr. 6, S. 53–59

[Marinschek et al. 2006] Marinschek, Martin ; Müllan, Gerald ; Schnabl, Andrea: *JSF @ Work. JavaServer Faces und Apache MyFaces erfolgreich einsetzen*. Dpunkt Verlag, 2006

[McClure et al. 2006] McClure, Wallace B. ; Cate, Scott ; Glavich, Paul ; Shoemaker, Craig: *Beginning Ajax with ASP.NET (Beginning)*. Birmingham, UK, UK : Wrox Press Ltd., 2006

[Microsoft Corporation 2007a] Microsoft Corporation (Hrsg.): *Windows Vista User Experience Guidelines*. 2007. – URL http://download.microsoft.com/download/e/1/9/e191fd8c-bce8-4dba-a9d5-2d4e3f3ec1d3/ux%20guide.pdf. – Letzter Zugriff am 17.02.2008

[Microsoft Corporation 2007b] Microsoft Corporation (Hrsg.): *Windows User Experience Guidelines. Windows User Experience Guidelines for Windows XP and Windows 2000*. 2007. – URL http://www.ms2.cn/downloads/details.aspx?familyid=B996E1E7-A83A-4CAE-936B-2A9D94B11BC5&displaylang=en. – Letzter Zugriff am 18.02.2008

[Moritz 2008] Moritz, Florian: *Rich Internet Applications (RIA): A Convergence of User Interface Paradigms of Web and Desktop Exemplified by JavaFX*, Fachhochschule Kaiserslautern,

Diplomarbeit, Januar 2008. – URL http://www.flomedia.de/diploma/documents/DiplomaThesisFlorianMoritz.pdf. – Letzter Zugriff am 16.06.2008

[Net Applications 2008] Net Applications (Hrsg.): *Browser Market Share - January, 2008 to March, 2008*. 2008. – URL http://marketshare.hitslink.com/report.aspx?qprid=0&qptimeframe=M&qpsp=108&qpnp=3&qpmr=100&qpdt=1&qpct=3&qpf=1. – Letzter Zugriff am 12.06.2009

[Oracle 2008] Oracle (Hrsg.): *Web User Interface Developer's Guide for Oracle Application Development Framework, 11g Release 1: DRAFT 5/1/08*. 2008. – URL http://download.oracle.com/otn_hosted_doc/jdeveloper/11/doc/b31973.pdf. – Letzter Zugriff am 31.05.2008

[SYS-CON Media Inc. 2008]SYS-CON Media Inc. (Hrsg.): *ComponentArt Releases Web.UI 2008.1: Featuring the New Hyper-Responsive AJAX UI*. 2008. – URL http://ajax.sys-con.com/read/529114_p.htm. – Letzter Zugriff am 11.05.2008

[Telerik 2008] Telerik (Hrsg.): *Why Choose RadControls for ASP.NET AJAX*. 2008. – URL http://www.telerik.com/products/aspnet-ajax/why.aspx. – Letzter Zugriff am 23.05.2008

[Tukey 1977] Tukey, John W.: *Exploratory Data Analysis (Addison-Wesley Series in Behavioral Science)*. 1. Addison Wesley, 1977

[Vogel et al. 2005] Vogel, Oliver ; Arnold, Ingo ; Chughtai, Arif ; Völter, Markus: *Software-Architektur. Grundlagen - Konzepte - Praxis*. Spektrum Akademischer Verlag, 2005

[W3C 2008] W3C (Hrsg.): *Rich Web Clients Activity Statement*. 2008. – URL http://www.w3.org/2006/rwc/Activity.html. – Letzter Zugriff am 16.06.2008

[Wenz 2007] Wenz, Christian: *Programming ASP .NET AJAX*. 1. O'Reilly Media, 2007

Abbildungsverzeichnis

Anlagenverzeichnis

Im Folgenden sind die Detailergebnisse der Evaluierung von eGovWDF angeführt. Dabei gilt folgende übergreifende Legende, falls nicht anders angegeben:

- Framework 1: ASP.NET 3.5 (Standard)
- Framework 2: AJAX Control Toolkit
- Framework 3: ComponentArt Web.UI
- Framework 4: NetAdvantage for ASP.NET
- Framework 5: Telerik Controls for ASP.NET AJAX
- Framework 6: DevExpress ASPxperience
- Framework 7: JSF (Standard)
- Framework 8: MyFaces Tomahawk
- Framework 9: JBoss RichFaces
- Framework 10: Oracle ADF Faces Rich Client
- Framework 11: NetAdvantage for JSF
- **Framework 12: eGovWDF for ASP.NET**

Falls nicht anders angegeben, werden auf der Ordinatenachse die Erfüllungsgrade (Gesamtanforderungserfüllungsgrad, Grad der JS-Unabhängigkeit bzw. Komponentenabdeckungsgrad) aufgetragen. Zusätzlich werden anlagenspezifische Legendendaten unter der jeweiligen Anlage vermerkt.

Falls nicht explizit anders angegeben, berücksichtigen die statistischen Kennzahlen Median, oberes Quartil, unteres Quartil, Minimum und Maximum sowie das arithmetische Mittel der Anforderungserfüllung stets alle in Kern (2008a) untersuchten Frameworks **mit Ausnahme von eGovWDF**. Diese Festlegung dient der Verhinderung von Messwertverzerrungen, da der aktuelle Stand der Technik unabhängig von eGovWDF als Vergleichsgrundlage für die Analyse von eGovWDF berechnet werden soll. Bei der graphischen Gegenüberstellung der verschiedenen Frameworks zum Vergleich der jeweiligen Anforderungserfüllungsgrade, wird eGovWDF jedoch miteingezeichnet, um einen direkten Vergleich zu erhalten. Die Ermittlung von Median, oberes Quartil, unteres Quartil, Minimum und Maximum erfolgt neben der Berechnung für jede einzelne Anforderung auch bezogen auf die Anforderungsart und bezogen auf den Komponententyp.

		Komponenten (Relative Anforderungserfüllung)								Notw. Vorauss. (relativ)	
Nr.	Framework	Dial-oge	Register-karten	Menüs	Symbol-leisten	Hierarch. Komp.	Fortschritts-anzeige	Infotips	Anforderungs-erfüllungsgrad	JS-Unabhängig-keit (auf Komp.)	Komp.-ab-deckung
1	ASP.NET 3.5 (Standard)	0,00	0,00	0,62	0,00	0,60	0,72	0,00	0,28	0,00	0,43
2	AJAX Control Toolkit	0,49	0,61	0,58	0,00	0,57	0,72	0,62	0,51	0,00	0,86
3	ComponentArt Web.UI	0,49	0,66	0,65	0,65	0,67	0,72	0,58	0,63	0,00	1,00
4	NetAdvantage for ASP.NET	0,54	0,62	0,52	0,65	0,67	0,72	0,00	0,53	0,00	0,86
5	Telerik Controls f. ASP.NET AJAX	0,57	0,66	0,68	0,69	0,66	0,72	0,00	0,57	0,00	0,86
6	DevExpress ASPxperience	0,56	0,63	0,65	0,60	0,60	0,72	0,69	0,64	0,00	1,00
7	JSF (Standard)	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
8	MyFaces Tomahawk	0,54	0,89	0,59	0,65	0,63	0,00	0,62	0,56	0,14	0,86
9	JBoss RichFaces	0,54	0,63	0,62	0,92	0,60	0,00	0,62	0,56	0,14	0,86
10	Oracle ADF Faces Rich Client	0,40	0,49	0,48	0,65	0,50	0,00	0,51	0,43	0,14	0,86
11	NetAdvantage for JSF	0,51	0,62	0,65	0,65	0,63	0,00	0,00	0,44	0,00	0,71
12	eGovWDF for ASP.NET	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Median (ohne eGovWDF)	0,51	0,62	0,62	0,65	0,60	0,72	0,51			
	O. Quartil (ohne eGovWDF)	0,54	0,64	0,65	0,65	0,65	0,72	0,62			
	Maximum (ohne eGovWDF)	0,57	0,89	0,68	0,92	0,67	0,72	0,69			
	Minimum (ohne eGovWDF)	0,00	0,00	0,00	0,00	0,00	0,00	0,00			
	U. Quartil (ohne eGovWDF)	0,44	0,55	0,55	0,30	0,59	0,00	0,00			

Kennzahlen

JS-Unabhängigkeit	Proz. Anteil der Frameworkkomponenten, die auch ohne JavaScript zumindest ihre Kernfunktionalität wahren
Komponentenabdeckung	Proz. Anteil d. im Rahmen d. Untersuchung geforderten Komponenten, die Bestandteil des Frameworks sind
Durchschnitt. Anforderungserfüllung	Durchschnittliche Anforderungserfüllung bei gleichgewichteten Anforderungen

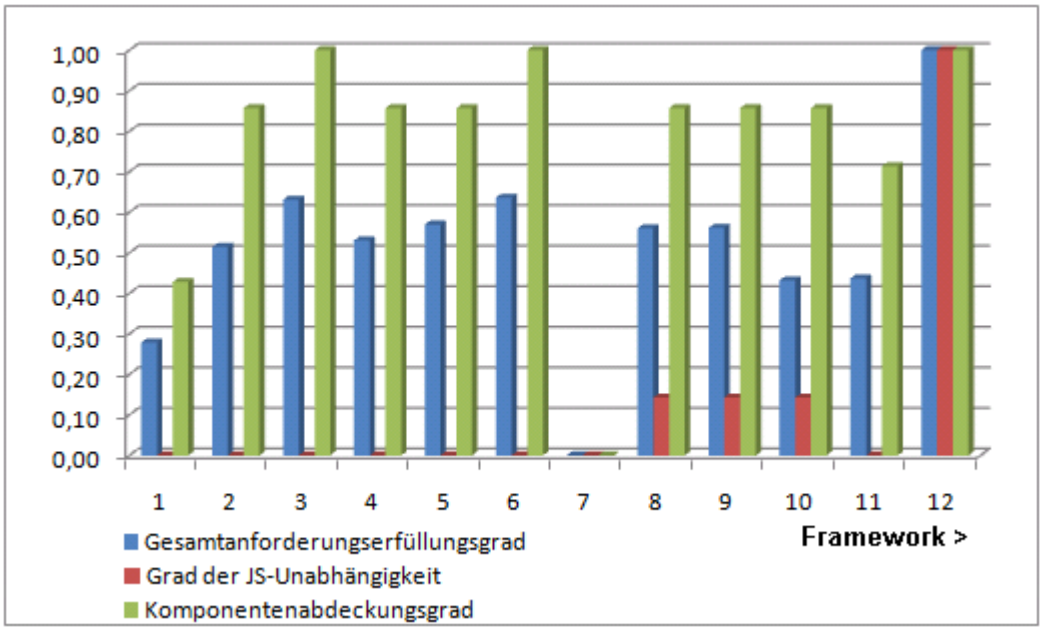
Notwendige Anforderungen

Durchschnittliche Anforderungserfüllung > 70%
JS-Unabhängigkeit > 90%
Komponentenabdeckung > 90%

Bemerkung

In obiger Darstellung wird im Schnittfeld von Framework und Komponente jeweils der erreichte Anforderungserfüllungsgrad kummuliert über alle Anforderungen der jeweiligen Framework-Komponente dargestellt.

A2. Vergleich der Primärkennzahlen von aktuellen Frameworks und eGovWDF



Anforderungserfüllung

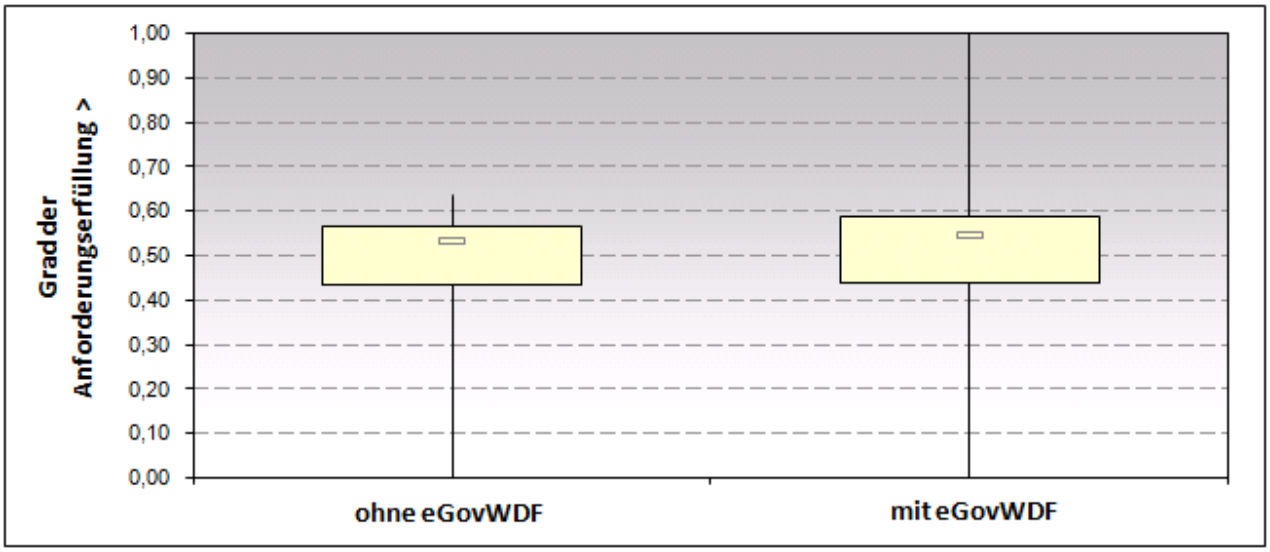
- Durchschnittliche Anforderungserfüllung (ohne eGovWDF): **0,47**
- Maximale Anforderungserfüllung (ohne eGovWDF): **0,64**

JS-Unabhängigkeit

- 3 von 11 Frameworks berücksichtigen JS-Unabhängigkeit
- Bei diesen 3 Frameworks liegt bei 14% der Komponenten JavaScript-Unabhängigkeit vor
- **Insgesamt liegt bei (14% von 3/11) = 4% aller Framework-Komponenten JS-Unabhängigkeit vor**

eGovWDF for ASP.NET erreicht als einziges Framework 100% der max. Anforderungserfüllung und 100% JS-Unabhängigkeit. Die Komponentenabdeckung liegt bei 100%.

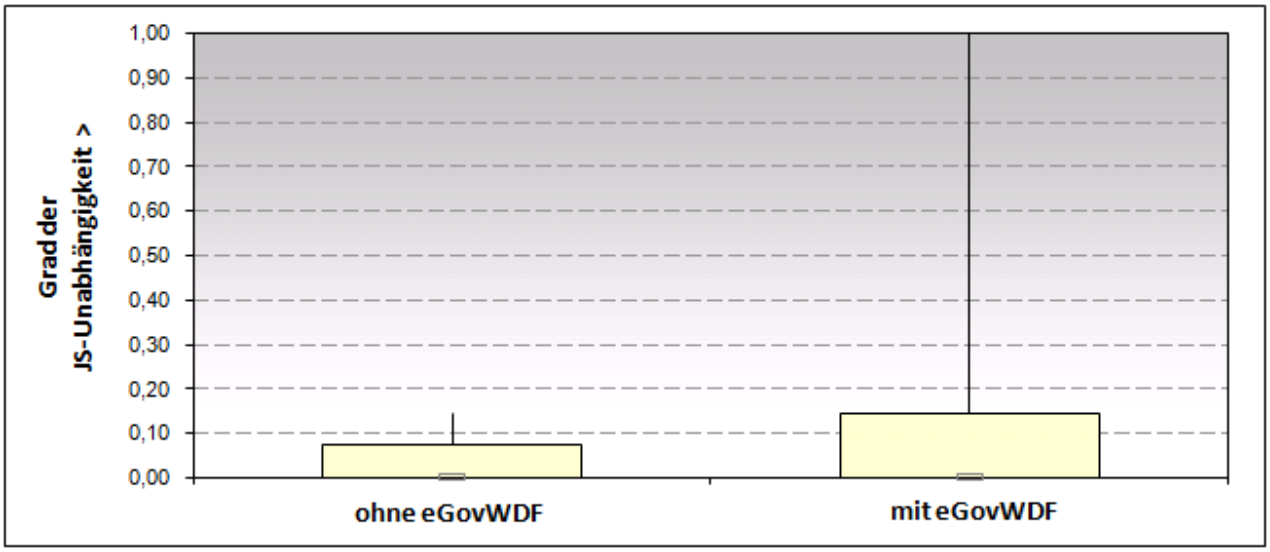
A3. Boxplot-Visualisierung der Gesamtanforderungserfüllung



Gesamtanforderungserfüllung

	ohne eGovWDF	mit eGovWDF
Median	0,53	0,55
O. Quartil	0,57	0,59
Maximum	0,64	1,00
Minimum	0,00	0,00
U. Quartil	0,43	0,44

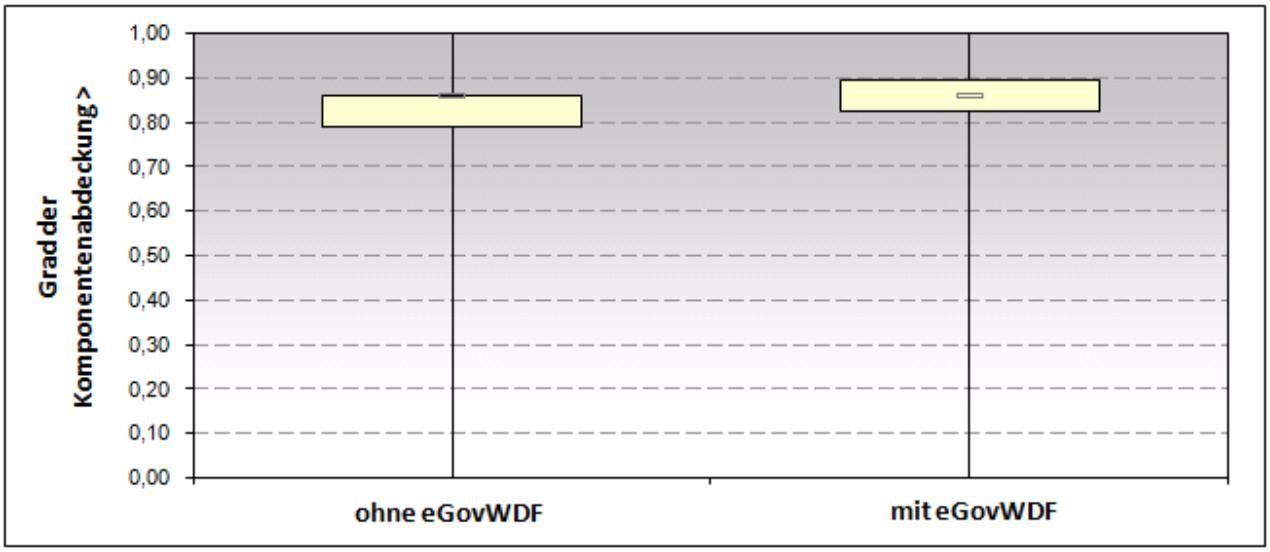
A4. Boxplot-Visualisierung der JS-Unabhängigkeit



JS-Unabhängigkeit

	ohne eGovWDF	mit eGovWDF
Median	0,00	0,00
O. Quartil	0,07	0,14
Maximum	0,14	1,00
Minimum	0,00	0,00
U. Quartil	0,00	0,00

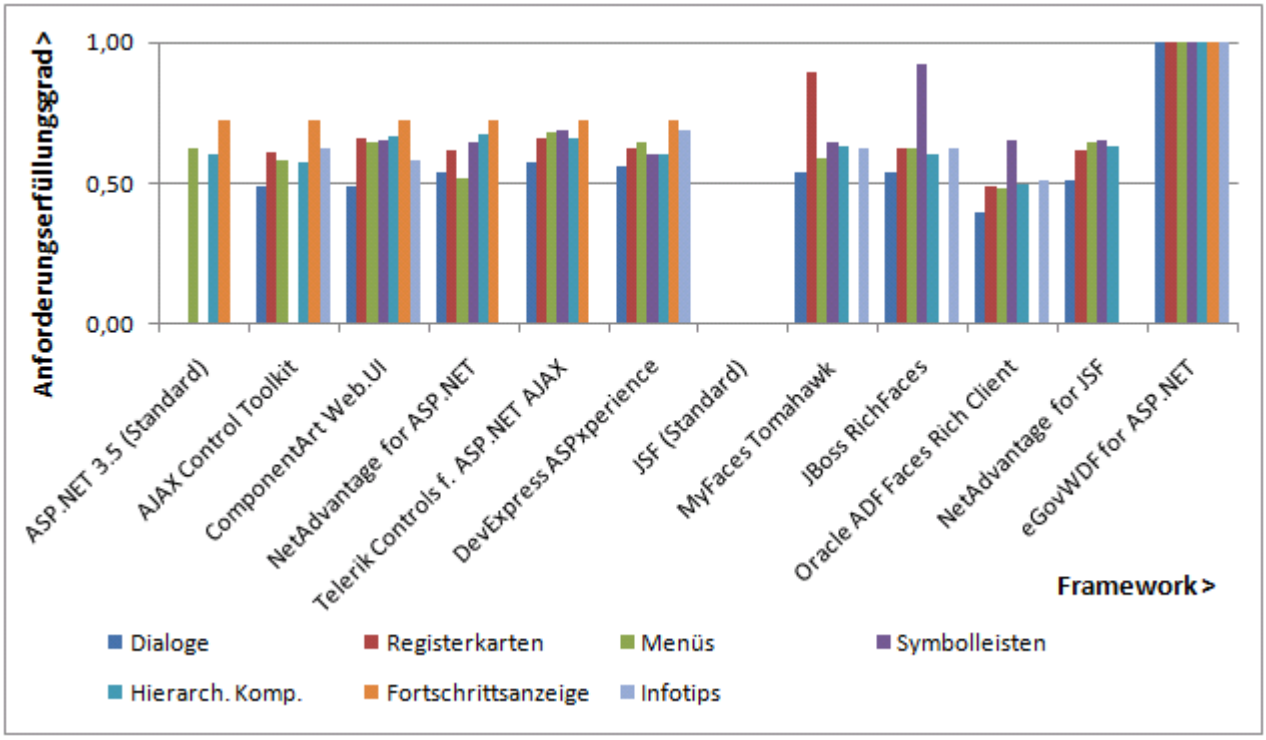
A5. Boxplot-Visualisierung der Komponentenabdeckung



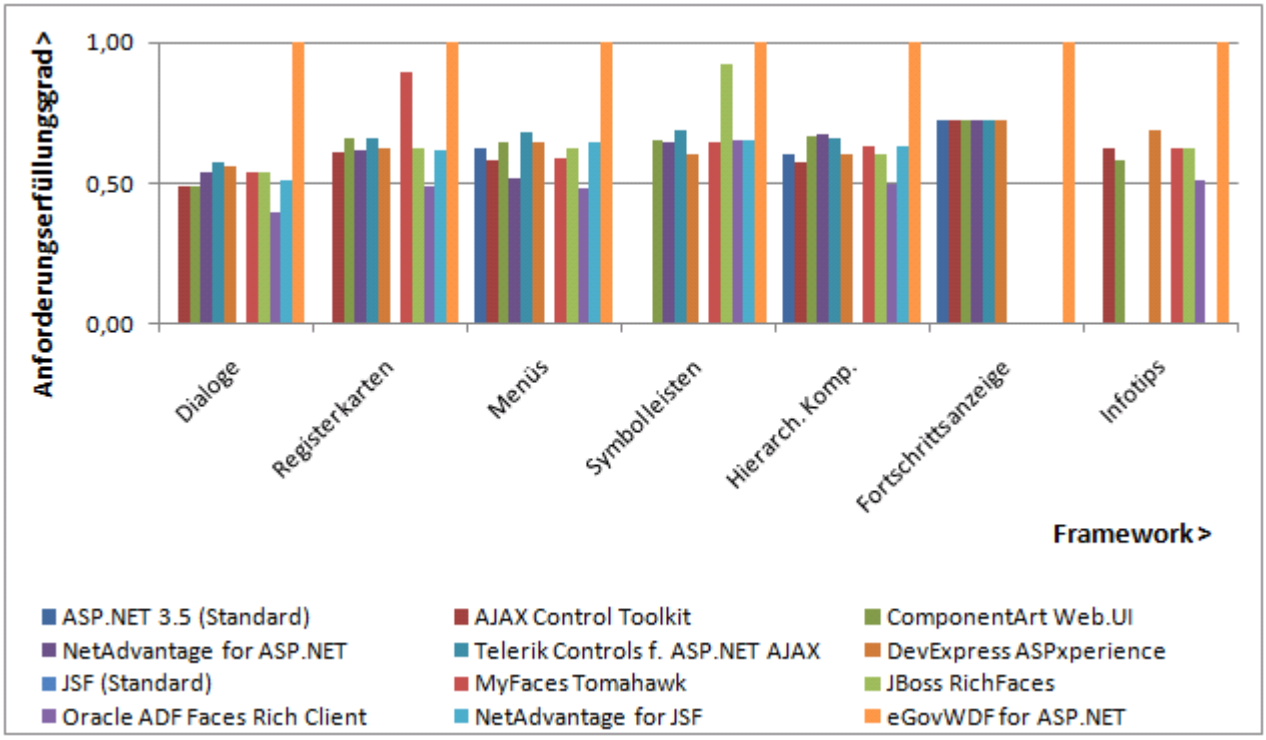
Komponentenabdeckung

	ohne eGovWDF	mit eGovWDF
Median	0,86	0,86
O. Quartil	0,86	0,89
Maximum	1,00	1,00
Minimum	0,00	0,00
U. Quartil	0,79	0,82

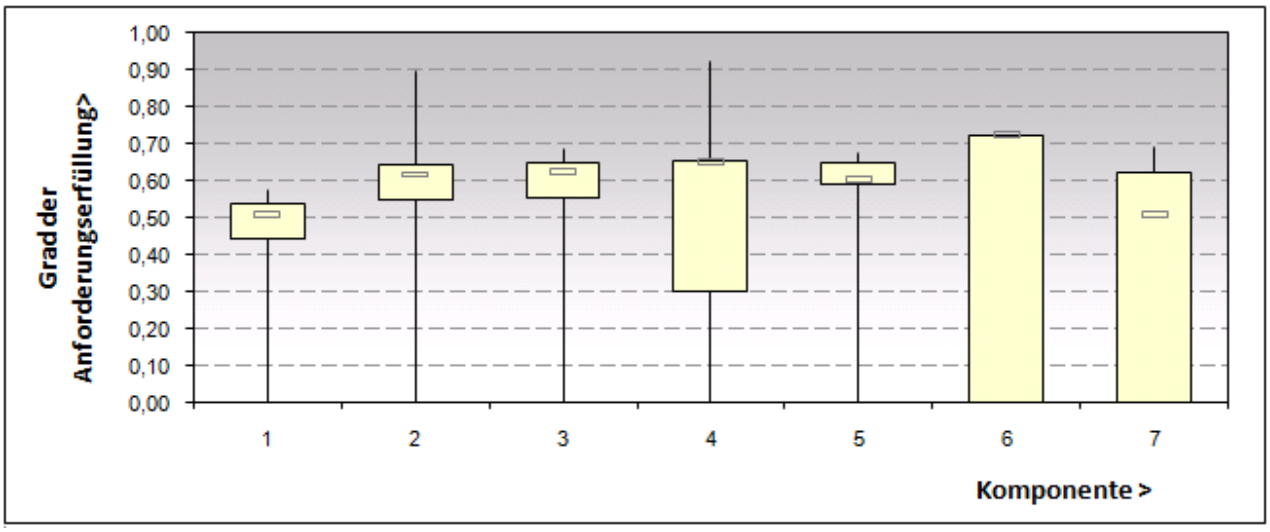
B1. Anforderungserfüllung pro Framework mit Komponenten als Reihen



B2. Anforderungserfüllung pro Komponente mit Frameworks als Reihen



B3. Boxplot-Visualisierung der Anforderungserfüllung nach Komponententyp



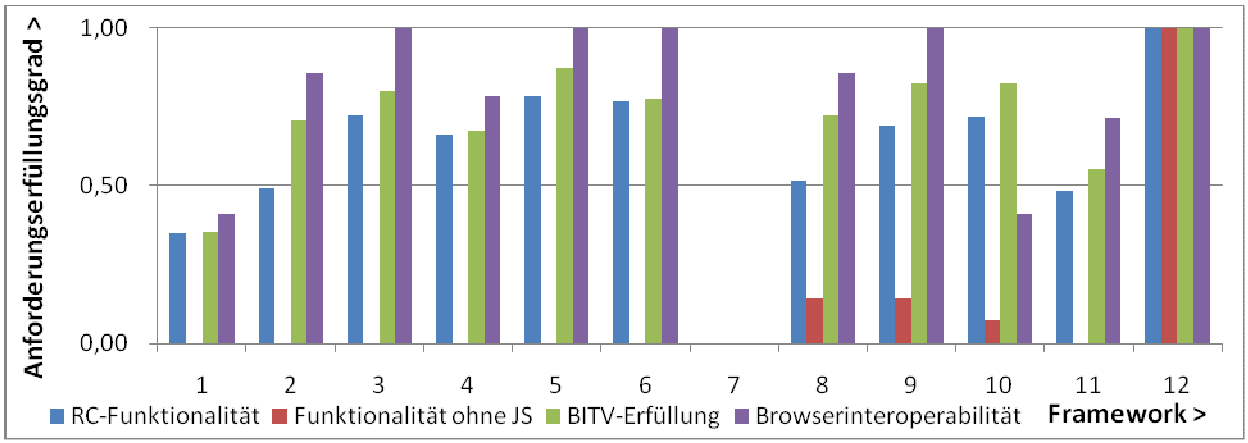
Legende:

- Komponente(ntyp) 1: Dialoge
- Komponente(ntyp) 2: Registerkartensteuerelemente
- Komponente(ntyp) 3: Menüs
- Komponente(ntyp) 4: Symbolleisten
- Komponente(ntyp) 5: Hierarchische Komponenten
- Komponente(ntyp) 6: Fortschrittsanzeige
- Komponente(ntyp) 7: Infotip-Komponenten

C1. Anforderungserfüllung nach Anforderungsarten

Nr.	Framework	RC-Funktionalität	Funkt. ohne JS	BITV	Browserinterop.
1	**ASP.NET 3.5 (Standard)**	0,35	0,00	0,35	0,41
2	**AJAX Control Toolkit**	0,49	0,00	0,71	0,86
3	**ComponentArt Web.UI**	0,72	0,00	0,80	1,00
4	**NetAdvantage for ASP.NET**	0,66	0,00	0,67	0,79
5	**Telerik Controls f. ASP.NET AJAX**	0,79	0,00	0,87	1,00
6	**DevExpress ASPxperience**	0,77	0,00	0,78	1,00
7	**JSF (Standard)**	0,00	0,00	0,00	0,00
8	**MyFaces Tomahawk**	0,52	0,14	0,72	0,86
9	**JBoss RichFaces**	0,69	0,14	0,83	1,00
10	**Oracle ADF Faces Rich Client**	0,72	0,07	0,83	0,41
11	**NetAdvantage for JSF**	0,48	0,00	0,55	0,71
12	**eGovWDF for ASP.NET**	1,00	1,00	1,00	1,00
	Mittelwert	0,56	0,03	0,65	0,73

C2. Anforderungserfüllung bezogen auf Frameworks und Anforderungsarten



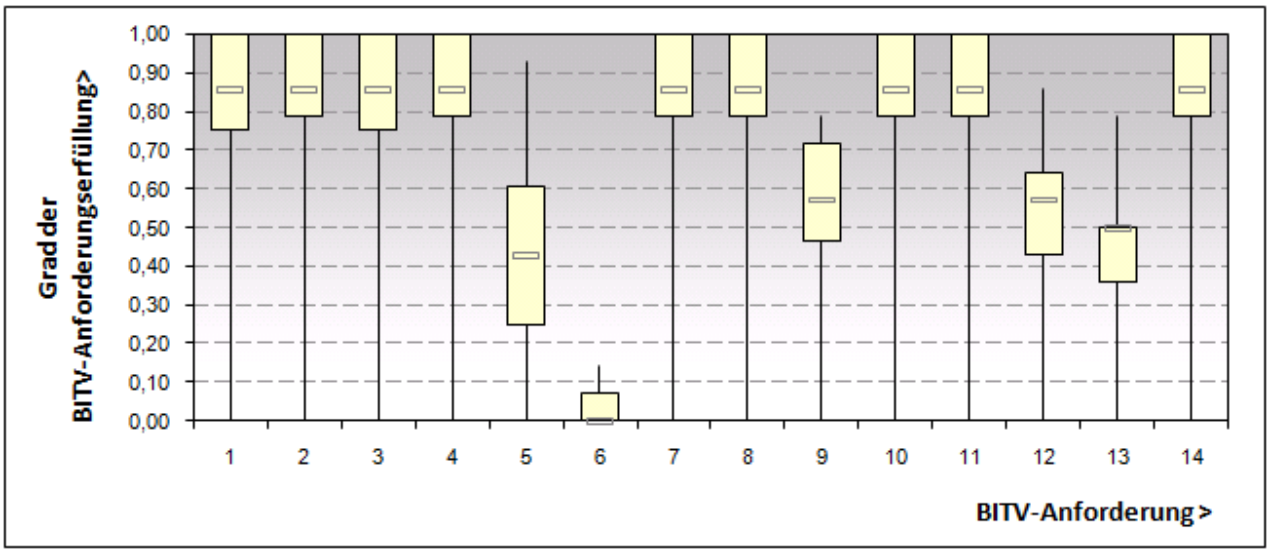
C3. BITV-Anforderungserfüllung

Nr.	Framework	A1	A2	A3	A4	A5	A6	A7
1	ASP.NET 3.5 (Standard)	0,43	0,43	0,29	0,43	0,29	0,00	0,43
2	AJAX Control Toolkit	0,86	0,86	0,79	0,86	0,79	0,00	0,86
3	ComponentArt Web.UI	1,00	1,00	1,00	1,00	0,43	0,00	1,00
4	NetAdvantage for ASP.NET	0,86	0,86	0,86	0,86	0,36	0,00	0,86
5	Telerik Controls f. ASP.NET AJAX	0,93	1,00	1,00	1,00	0,93	0,00	1,00
6	DevExpress ASPxperience	1,00	1,00	0,93	1,00	0,21	0,00	1,00
7	JSF (Standard)	0,00	0,00	0,00	0,00	0,00	0,00	0,00
8	MyFaces Tomahawk	0,86	0,86	0,86	0,86	0,57	0,14	0,86
9	JBoss RichFaces	1,00	1,00	1,00	1,00	0,50	0,14	1,00
10	Oracle ADF Faces Rich Client	1,00	1,00	1,00	1,00	0,64	0,07	1,00
11	NetAdvantage for JSF	0,64	0,71	0,71	0,71	0,14	0,07	0,71
12	eGovWDF for ASP.NET	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Median (ohne eGovWDF)	0,86	0,86	0,86	0,86	0,43	0,00	0,86
	O. Quartil (ohne eGovWDF)	1,00	1,00	1,00	1,00	0,61	0,07	1,00
	Maximum (ohne eGovWDF)	1,00	1,00	1,00	1,00	0,93	0,14	1,00
	Minimum (ohne eGovWDF)	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	U. Quartil (ohne eGovWDF)	0,75	0,79	0,75	0,79	0,25	0,00	0,79

Nr.	Framework	A8	A9	A10	A11	A12	A13	A14
1	ASP.NET 3.5 (Standard)	0,43	0,43	0,43	0,43	0,29	0,21	0,43
2	AJAX Control Toolkit	0,86	0,57	0,86	0,86	0,50	0,43	0,86
3	ComponentArt Web.UI	1,00	0,71	1,00	1,00	0,57	0,50	1,00
4	NetAdvantage for ASP.NET	0,86	0,50	0,86	0,86	0,43	0,43	0,86
5	Telerik Controls f. ASP.NET AJAX	1,00	0,71	1,00	1,00	0,86	0,79	1,00
6	DevExpress ASPxperience	1,00	0,64	1,00	1,00	0,57	0,50	1,00
7	JSF (Standard)	0,00	0,00	0,00	0,00	0,00	0,00	0,00
8	MyFaces Tomahawk	0,86	0,57	0,86	0,86	0,64	0,50	0,86
9	JBoss RichFaces	1,00	0,79	1,00	1,00	0,64	0,50	1,00
10	Oracle ADF Faces Rich Client	1,00	0,71	1,00	1,00	0,64	0,50	1,00
11	NetAdvantage for JSF	0,71	0,43	0,71	0,71	0,43	0,29	0,71
12	eGovWDF for ASP.NET	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Median (ohne eGovWDF)	0,86	0,57	0,86	0,86	0,57	0,50	0,86
	O. Quartil (ohne eGovWDF)	1,00	0,71	1,00	1,00	0,64	0,50	1,00
	Maximum (ohne eGovWDF)	1,00	0,79	1,00	1,00	0,86	0,79	1,00
	Minimum (ohne eGovWDF)	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	U. Quartil (ohne eGovWDF)	0,79	0,46	0,79	0,79	0,43	0,36	0,79

Legende: A1 bis A14 repräsentieren die korrespondierenden Anforderungen der BITV

C4. Boxplot-Visualisierung der BITV-Anforderungserfüllung



Legende: A1 bis A14 repräsentieren die korrespondierenden Anforderungen der BITV

D. Detailergebniswerte der Untersuchung von eGovWDF

Prüfaspekt	Prüfsubaspekt	Dialoge	Registerkarten	Menüs	Symbolleisten	Hierarch. Steuerelemente	Fortschrittsanzeige	Infotips	Über alle Komp. gemittelte Ges.anf.erf.
Rich Client-Funktionalität	Request-Lebenszyklus-Persistenz	1,00	1,00		1,00	1,00			
	Anpassbarkeit	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Verhalten	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Entwicklungsunterstützung	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Wiederverwendbarkeit (ext. Res.)	1,00							
	Definierter Datenaustauschprozess	1,00							
	Standard-Dialoge	1,00							
	Durchschnitt	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Funktionsweise bei nicht verfügbarem JavaScript	Funktionsweise bei nicht verfügbarem JavaScript	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Durchschnitt	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
BITV-Anforderungserfüllung	Anforderung 1	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 2	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 3	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 4	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 5	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 6	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 7	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 8	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 9	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 10	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 11	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 12	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 13	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
	Anforderung 14	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00

	Durchschnitt	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Browserinter-operabilität	Internet Explorer 6.0+	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Firefox 2.0+	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Opera 9	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Safari 3.1	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
	Durchschnitt	1,00	1,00	1,00	1,00	1,00	1,00	1,00	1,00
Gesamt	**Durchschnitt**	1,00	1,00	1,00	1,00	1,00	1,00	1,00	
								1,00	