Methoden zur intuitiven Prozessmodellierung in Webportalen

Diplomarbeit
am Lehrstuhl Software Engineering
Prof. Dr. Lutz Prechelt
Institut für Mathematik/Informatik
Freie Universität Berlin

vorgelegt von
Julia Schenk

Betreuer: Prof. Dr. Lutz Prechelt

Julia Schenk
Matrikelnummer: 3482271

Inhaltsverzeichnis

Abbildungsverzeichnis **VII**

Tabellenverzeichnis **X**

Auflistungen **XI**

1 Einleitung **1**
1.1 Motivation 1
1.2 Ziel 2

2 Grundlagen und Einordnung der Arbeit **3**
2.1 Begriffliche Einführung 4
2.2 Mentale Aspekte der Programmierung 6
2.2.1 Psychologie des Programmierens 7
2.2.2 Human Computer Interaction (HCI) 9
2.2.3 Das Framework der kognitiven Dimensionen 10
2.3 Vergleich textueller und visueller Programmiersprachen 13
2.4 Konzepte visueller Programmiersprachen 18
2.4.1 Steuerflussorientierte visuelle Programmiersprachen 18
2.4.2 Datenflussorientierte visuelle Programmiersprachen 19
2.4.3 Funktionsorientierte visuelle Programmiersprachen 20
2.4.4 Objektorientierte visuelle Programmiersprachen 21
2.4.5 Constraintorientierte visuelle Programmiersprachen 22
2.4.6 Regelorientierte visuelle Programmiersprachen 22
2.4.7 Beispielorientierte visuelle Programmiersprachen 23
2.4.8 Formularorientierte visuelle Programmiersprachen 24
2.4.9 Multiparadigmenorientierte visuelle Programmiersprachen 24

3 Evaluierung bestehender Systeme **27**
3.1 Beschreibung der Problemdomäne 28
3.2 Bestehende Systeme 35
3.3 Produkte für die Usability-Tests 35

4 Durchführung der Usability-Tests **37**
4.1 Konzepte von Usability-Tests 38

4.1.1 Interaktionsaufzeichnung . . . 38
4.1.2 Videobeobachtung . . . 38
4.1.3 Blickbewegungsregistrierung (Eye Tracking) . . . 39
4.1.4 Lautes Denken (Thinking Aloud) . . . 39
4.1.5 Usability-Inspektionsmethoden . . . 39
4.1.6 Moderierter Gruppentest . . . 40
4.2 Auswahl einer Usablility-Testmethode . . . 40
4.3 Anzahl der Probanden . . . 41
4.4 Das Usability-Labor . . . 43
4.5 Der Testplan . . . 44
4.6 Durchführungsbeschreibung . . . 47
4.7 Inhalte der Usability-Tests . . . 48
4.7.1 *jotform* Formulargenerator . . . 48
4.7.2 *formdesk* Workflowmodellierung . . . 51
4.7.3 *WebDSL* . . . 54

5 Auswertung und Analyse der erhobenen Testdaten 57
5.1 Quantifizierung der aufgetretenen Usability-Probleme . . . 61
5.2 Mittelwert der aufgetretenen Usability-Probleme pro Mikrotask . . . 65
5.3 Analyse der aufgetretenen Usability-Probleme . . . 69
5.3.1 Analyse für den *jotform* Formulargenerator . . . 70
5.3.1.1 Anzahl der Usability-Probleme pro Mikrotask . . . 72
5.3.1.2 Gruppierung der Usability-Probleme nach Problemkategorie . 74
5.3.1.3 Auswertung der Interviews . . . 76
5.3.2 Analyse für das *formdesk* Formular-Management-System . . . 78
5.3.2.1 Anzahl der Usability-Probleme pro Mikrotask . . . 80
5.3.2.2 Gruppierung der Usability-Probleme nach Problemkategorie . 83
5.3.2.3 Auswertung der Interviews . . . 85
5.3.3 *WebDSL* . . . 87
5.4 Fazit . . . 88
5.4.1 *jotform* . . . 88
5.4.2 *formdesk* . . . 89
5.4.3 *webdsl* . . . 90
5.4.4 Schlussfolgerungen . . . 90

6 Produktvision 93
6.1 Funktionale Anforderungen . . . 94
6.1.1 Exemplarisch umzusetzende funktionale Anforderungen . . . 94
6.1.2 Im Prototypen konzeptionell vorzusehende funktionale Anforderungen 95
6.2 Deklarative Beschreibung der Produkteigenschaften . . . 96

7 Das konzeptuelle Modell des Benutzerinterfaces 99

7.1 Elemente des schrittweisen Modellierungsansatzes 102
7.2 Reihenfolge der identifizierten Schritte 103
7.3 Darstellung des schrittweisen Modellierungsansatzes 105
7.4 Konzeptueller Aufbau der einzelnen Schritte 111
7.4.1 Auswahl des Layouts 111
7.4.2 Bearbeitung des Formulars und der Formularelemente 111
7.4.2.1 Untersuchungen zum Bedienkonzept unter Berücksichtigung verschiedener Benutzertypen 118
7.4.3 Bearbeitung der E-Mail-Berichte und der Formulardaten 121
7.4.4 Hochladen des Formulars 129

8 Technische Realisierung 131
8.1 Verwendete Technologien 131
8.2 Architektur 132
8.2.1 Models 133
8.3 Weitere Aspekte der Umsetzung 138
8.4 Qualitätssicherung 140

9 Verifikation der Überlegungen 145
9.1 Durchführung von Usability-Tests mit der entwickelten Methode 146

10 Analyse der durchgeführten Usability-Tests mit der entwickelten Methode 149
10.1 Mittelwert der aufgetretenen Usability-Probleme pro Mikrotask 151
10.2 Anzahl der Usability-Probleme pro Problemkategorie 157
10.3 Gruppierung der Usability-Probleme nach Problemkategorie 159
10.4 Auswertung der Interviews 160

11 Fazit 163

Literaturverzeichnis XII

Anhänge XV

A Kategorisierte Übersicht der betrachteten Produkte XV

B Screenshots betrachteter Produkte XVII

C Der Testplan XX

D Screenshots der getesteten Produkte XLIII

E Auswertungstabelle der Testdurchführungen XLIX
Tabellarische Auswertung der Testdurchführungen mit der Anwendung *jotform* . . . XLIX
Tabellarische Auswertung der Testdurchführungen mit der Anwendung *formdesk* . LX

Tabellarische Auswertung der Testdurchführungen mit der Anwendung *perForm* . . LXXIII
Grafische Darstellung der Auswertungtabelle für das Produkt *jotform* LXXVII
Grafische Darstellung der Auswertungtabelle für das Produkt *formdesk* LXXVIII
Grafische Darstellung der Auswertungtabelle für das Produkt *perForm* LXXIX

Abbildungsverzeichnis

2.1 Mengendarstellung visueller und textueller Programmiersprachen 4
2.2 Übersicht visueller Programmierkonzepte 26

3.1 User Story ‘Anmeldung zu einer Lehrveranstaltung’ in BPMN 30
3.2 User Story ‘externe Anfrage über ein Online-Formular’ in BPMN 32
3.3 User Story ‘interne Anfrage über ein Online-Formular’ in BPMN 34

4.1 Problemfindungskurve bei Usability-Tests 41
4.2 Das Usability-Labor 43
4.3 Repräsentation eines Tasks im Testplan 46

5.1 *jotform*: Umsetzungsprobleme der einzelnen Probanden 64
5.2 *jotform*: Mittelwert der Usability-Probleme pro Mikrotask 65
5.3 *formdesk*: Mittelwert der Usability-Probleme pro Mikrotask 67
5.4 *jotform*: Summe aufgetretener Usability-Probleme pro Problemkategorie . . . 74
5.5 *formdesk*: Summe aufgetretener Usability-Probleme pro Problemkategorie . 83

7.1 Vorgehen bei der Entwicklung des Wireframes 101
7.2 Auswahl einer Dokumentenvorlage in *Pages* 105
7.3 Darstellung der Modellierungsschritte durch verbundene Steuerelemente . . . 107
7.4 Darstellung der Modellierungsschritte durch farbige, verbundene Interaktionselemente 107
7.5 Darstellung der Modellierungsschritte durch Registerkarten 108
7.6 Darstellung der Modellierungsschritte mit Zahlen 108
7.7 Breadcrumb-Navigation des Internetportals der Freien Universität Berlin . . . 109
7.8 Darstellung der Modellierungsschritte ähnlich Breadcrumb-Navigation 109
7.9 Darstellung der Modellierungsschritte ähnlich Breadcrumb-Navigation mit farbiger Nummerierung 110
7.10 Anordnung des Eigenschaften-Inspektors unten rechts 113
7.11 Anordnung des Eigenschaften-Inspektors oben rechts 115
7.12 Wechselnde Anzeige von Eigenschaften-Inspektor und Formularelementen . . 117
7.13 Hinzufügen von Regeln in dem E-Mail Client *Apple Mail* 121
7.14 Default-Ansicht der E-Mail-Bestätigungen in einem Dialogfenster 123
7.15 Erweiterte Ansicht der E-Mail-Bestätigungen in einem Dialogfenster 124
7.16 Bearbeitung des Inhalts der E-Mail-Bestätigungen bei mehreren Dialogfenstern 126

7.17 Bearbeitung der Versandkriterien der E-Mail-Bestätigungen bei mehreren Dialogfenstern . . . 128

8.1 *perForm*: Datenbanktabellen . . . 136
8.2 *perForm*: Auswahl eines Formularelements bei einem Anzeigekriterium . . . 138
8.3 *perForm*: Bedingungen für Formularelemente mit fest definierten Werten . . . 139
8.4 *perForm*: Bedingungen für Formularelemente mit Texteingaben . . . 139
8.5 *perForm*: Werte für Formularelemente mit fest definierten Werten . . . 140
8.6 *perForm*: Werteingabe für Bedingung eines Anzeigekriteriums mit Texteingaben 140
8.7 *perForm*: Definition von Benachrichtigungskriterien . . . 140
8.8 Das spec-Verzeichnis in Kontext einer *Ruby on Rails* Anwendung . . . 142
8.9 *RSpec*: Controller-Verhalten bei einem Drop-Event . . . 143
8.10 Ausgabe der *RSpec* Tests bei Erfüllung des spezifizierten Verhaltens . . . 144

10.1 *perForm*: Mittelwert der Usability-Probleme pro Mikrotask . . . 151
10.2 *perForm*: Hinweis zu Bedienungskonzepten . . . 152
10.3 *perForm*: Bearbeitungsmodus von Formularelementen . . . 153
10.4 *perForm*: Anzeigeoptionen . . . 154
10.5 *perForm*: E-Mail-Bestätigung verfassen . . . 155
10.6 *perForm*: Benachrichtigungskriterien . . . 155
10.7 *perForm*: Summe aufgetretener Probleme pro Problemkategorie . . . 159
10.8 *perForm*: Palette der Formularelemente . . . 162

11.1 Summe aller in Usability-Tests aufgetretener Probleme pro Problemkategorie . 164
11.2 Summe aufgetretener Usability-Probleme pro Anwendung . . . 165

B.1 BizFlow Formulargenerator . . . XVII
B.2 WorkflowXPert . . . XVIII
B.3 Savvion Process Modeler . . . XVIII
B.4 PageBreeze HTML Editor . . . XIX
B.5 4Webmaster Formulargenerator . . . XIX

D.1 *jotform*: Startseite . . . XLIII
D.2 *jotform*: Wizard . . . XLIV
D.3 *formdesk*: Übersicht über die Anwendung . . . XLIV
D.4 *formdesk*: Auszuwählende Formularelemente . . . XLV
D.5 *formdesk*: Bearbeitungsmodus eines Formularelements . . . XLV
D.6 *formdesk*: Schritt der Layoutauswahl . . . XLVI
D.7 *formdesk*: Allgemeine Formulareinstellungen . . . XLVII
D.8 *formdesk*: Unterschiedliche Berichte . . . XLVIII
D.9 *formdesk*: Bearbeiten des Inhalts eines E-Mail Berichts . . . XLVIII

E.1 *jotform*: Orientierungsprobleme der einzelnen Mikrotasks . . . LXXVII

E.2 *jotform*: Umsetzungs-Probleme der einzelnen Mikrotasks LXXVII
E.3 *formdesk*: Umsetzungsprobleme der einzelnen Mikrotasks LXXVIII
E.4 *formdesk*: Umsetzungs-Probleme der einzelnen Mikrotasks LXXVIII
E.5 *perForm*: Umsetzungsprobleme der einzelnen Mikrotasks LXXIX
E.6 *perForm*: Umsetzungs-Probleme der einzelnen Mikrotasks LXXIX

Tabellenverzeichnis

2.1 Gegenüberstellung visueller und verbaler Programmiersprachen 16

3.1 User Story 1: ‘Anmeldung zu einer Lehrveranstaltung’ 29
3.2 User Story 2: ‘externe Anfrage’ 31
3.3 User Story 3: ‘interne Anfrage’ 33

5.1 *jotform*: Mikrotasks 58
5.2 *formdesk*: Mikrotasks 60
5.3 *jotform*: Zentrale Aspekte der Mikrotasks 66
5.4 *formdesk*: Zentrale Aspekte der Mikrotasks 68
5.5 *jotform*: Aufgetretene Usability-Probleme 72
5.6 *jotform*: Aufgetretene Usability-Probleme nach Problemkategorie 73
5.7 *formdesk*: Aufgetretene Usability-Probleme 80
5.8 *formdesk*: Aufgetretene Usability-Probleme nach Problemkategorie 81

7.1 Objekte/Aktionen Analyse der Anwendungsdomäne 102
7.2 Alternierender Benutzertyp - Anzahl Mausklicks 119
7.3 Kontinuierlicher Benutzertyp - Anzahl Mausklicks 119

10.1 *perForm*: Mikrotasks 150
10.2 *perForm*: Aufgetretene Usability-Probleme 157
10.3 *perForm*: Aufgetretene Usability-Probleme nach Problemkategorie 157

Auflistungen

4.1 *WebDSL* Beispielcode 55
4.2 Einfache Formularstrukturen in *WebDSL* 56

8.1 Auszug aus dem Form-Model 135
8.2 handle_element_drop-Methode 137

1 Einleitung

1.1 Motivation

Die Änderung von Prozesslogik in Webportalen ist aufwendig und erfordert häufig einen Entwickler. Andererseits hat der Domänenexperte das Wissen über den optimalen Prozessablauf, kann dieses aber nicht direkt umsetzen, sondern muss es an einen Entwickler kommunizieren. Dabei kann es aufgrund von Verständnis- oder Kommunikationsproblemen zu einer nicht optimalen oder fehlerhaften Abbildung des Prozesses kommen.

Es existieren bereits zahlreiche Modellierungswerkzeuge zur Abbildung und Umsetzung von Geschäftsprozessen durch Domänenexperten, um so die semantische Lücke zwischen Entwickler und Domänenexperte zu umgehen und flexibel und kostensparend auf Geschäftsprozessänderungen reagieren zu können. Diese Werkzeuge sind für den vielfältigen und universellen Einsatz in großen Unternehmensstrukturen konzipiert, um dort komplexe Geschäftsprozesse und Workflows zu modellieren.

Die meisten am Markt befindlichen Produkte sind für den konkreten Einsatzzweck der Prozessmodellierung in Webportalen deutlich zu komplex und umfangreich. Dies liegt aber weniger an einer schlechten Usability der Produkte allgemein, sondern ist vielmehr im großen und unspezifischen Leistungsumfang dieser Produkte begründet. Somit stellt es eine sehr interessante Aufgabe dar, für das abgeschlossene Problemfeld der Prozessmodellierung in Webportalen eine Lösung zu finden, mit der einfache Prozesse in Webportalen von Domänenexperten unter geringem Lernaufwand leicht und intuitiv selbst modelliert und eingefügt werden können. Im Kontext dieser Domäne zählen aus Sicht der Domänenexperten das leichte und intuitive Definieren von HTML-Formularen sowie das Verknüpfen dieser Formulare mit bestimmten Funktionen und Abläufen zu den zentralen Anforderungen. Hinsichtlich dieser Anforderungen sind Funktionen wie eine Reminder-Funktion, ein Dispatcher-Mechanismus, eine Status- und Prioritätenvergabe, eine Stellvertreter- und Gruppenregelung sowie das Definieren von Eskalationsszenarien zu nennen.

1.2 Ziel

Für das Problemfeld der Prozessmodellierung in Webportalen wird eine Methode entwickelt, die es einem Domänenexperten ermöglicht, in intuitiver und leicht erlernbarer Form Prozessabläufe zu modellieren und einzubinden. Er kann so flexibel auf Änderungen in Prozessabläufen reagieren. Das System wird dadurch in seiner Stabilität nicht beeinträchtigt und nichtfunktionale Anforderungen werden berücksichtigt. Die Verifizierung dieser Methode wird anhand eines Prototypen erfolgen.

2 Grundlagen und Einordnung der Arbeit

Die in der Einleitung erwähnte Modellierung von Prozessen in Webportalen durch Domänenexperten, also End-Benutzer, welche nicht über das technische Wissen zur Programmierung von Online-Formularen in Verknüpfung mit Geschäftslogik verfügen, fällt in den Bereich des End User Development. In [Wul] wird hervorgehoben, dass alle Bereiche, in denen End-Benutzer Software entwickeln oder Änderungen an bestehender Software vornehmen, zum End User Development zählen.

In [Wul] wird in der Begriff End User Development folgendermaßen definiert:

"Der Begriff End User Development drückt aus, dass End-Benutzer ohne Programmierkenntnisse ihre Software selbst an ihre Bedürfnisse und Anforderungen anpassen und so aktiv ihre Software zur Laufzeit gestalten, verändern und ergänzen können. Das Ziel der End User Development Forschung ist es, Technologien und Methoden zu entwickeln, durch die es möglich wird, Software flexibler zu gestalten, so dass der End-Benutzer sie nicht nur leicht anwenden, sondern auch leicht an seine eigenen Bedürfnisse anpassen kann.".

Auf Basis dieser Definition wird die vorliegende Arbeit im Rahmen der End User Development Forschung eingeordnet.

Nach [Wul] muss der End-Benutzer für die Verwendung und Anpassung der Funktionen seiner Software in der Lage sein, *"auf einfache Weise die Flexibilisierungsmöglichkeiten seiner Software zu verstehen und erlernen zu können"*. Entsprechend ist es das Ziel dieser Arbeit, für die End-Benutzer der Domäne der einfachen Geschäftsprozesse in Webportalen, insbesondere dem Online-Formularwesen, eine Methode zu entwickeln, die sie leicht verstehen und mit der sie einfach und intuitiv ihre Anforderungen umsetzen können.

In [Wul] werden zwei Komplexitätsstufen des End User Developments unterschieden. In die untere Komplexitätsklasse fällt die Möglichkeit der Anpassung der Benutzeroberfläche durch den Benutzer. Der Benutzer kann, wie in vielen geläufigen Office-Anwendungen, selbst das Aussehen seiner Bedienungsoberfläche und die Anordnung der darauf enthaltenen Bedienelemente festlegen. Hat der End-Benutzer jedoch die Möglichkeit, selbst Software zu erstellen oder anzupassen, so fällt dieses in die nächst höhere Stufe der Komplexität im Bereich des End User Development. In dieser Komplexitätsklasse wird der Benutzer durch die Verwendung visueller oder textueller Programmierung oder Skriptsprachen zum Programmierer. Als Beispiel sei hier die Programmierung von Makros in Office-Anwendungen genannt.

2.1 Begriffliche Einführung

Eine einheitliche Terminologie für **visuelle Programmiersprachen** ist schwer zu finden, da das Schlagwort 'visuell' auch Systemen anhaftet, die wenig visuelle Sprachelemente benutzen, wie z.B. Programmierumgebungen, die zwar im Wesentlichen einen textuellen Editor verwenden und lediglich visuelle Zusatzfunktionen für beispielsweise den Entwurf grafischer Benutzeroberflächen bereit stellen. Bei diesen integrierten Programmierumgebungen (IDEs) erfolgt die Formulierung der Programmlogik in einer textuellen Programmiersprache und die grafischen Komponenten dienen meistens nur der Erstellung der Benutzerinterfaces [Scha].

In [Scha] wird die Problematik des Definitionsversuches für visuelle Programmierung dadurch begründet, dass der Begriff 'visuell' im Allgemeinen die Bedeutung 'das Sehen oder den Gesichtssinn betreffend' hat. Dieses allgemeine Verständnis des Begriffes bezeichnet also eine bestimmte Wahrnehmungsart des Menschen und bezieht sich somit nicht nur auf grafischen Elemente, sondern auch auf normalen Text, da dieser genauso wie grafische Elemente sichtbar ist. Dementsprechend ist der Komplementärbegriff zu **'visuell'** auch nicht 'textuell', sondern **'verbal'**.

Diese Abgrenzung der Begriffe wird auch den Begrifflichkeiten der Kognitionspsychologie gerecht, bei der bei den informationsverarbeitenden Prozessen des Menschen nach Verarbeitung visueller und verbaler Informationen unterschieden wird. Die Bezeichnung 'verbale Information' kann also verstanden werden als 'durch Worte Repräsentierte Information', im Gegensatz zu rein visueller, also anhand von Grafiken und Piktogrammen dargestellter Information.

Entsprechend dieser Differenzierung verbaler und visueller Sprachen lassen sich die textuellen Sprachen als Teilklasse der visuellen Sprachen verstehen, was in Abbildung 2.1 als Mengendarstellung veranschaulicht ist.

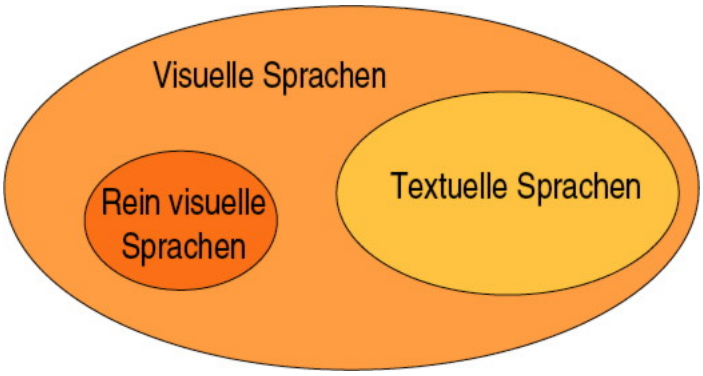


Abbildung 2.1: Mengendarstellung visueller und textueller Programmiersprachen, Quelle:[Schb]

In Anlehnung an diese Einordnung visueller und verbaler Sprachen werden in [Schb] die beiden folgenden Implementierungsstrategien unterschieden.

- textuell (verbale Repräsentation der Informationen): Universaleditor + Parsing
- rein visuell (grafische Repräsentation der Informationen): Struktureditor

In diesem Kontext der Programmierung bedeutet 'visuell', dass mindestens eine unverzichtbare Information über das betrachtete Objekt, welches Teil eines Programms ist, nur durch das visuelle Wahrnehmungssystem des Menschen gewonnen werden kann [Scha]. Visuell kann auch als **'visuell informativ'** verstanden werden, da zum Erhalt der Informationen des Objekts das visuelle Wahrnehmungssystem in diesem Sinne ein unverzichtbares Instrumentarium ist. Rein textuell repräsentierte Informationen sind zwar auch sichtbar, aber man kann die repräsentierten Daten nicht nur über das Auge, sondern ohne Informationsverlust auch über das Gehör und andere Sinne aufnehmen. Daher sind textuelle Repräsentation von Objekten zwar visuell, aber nicht visuell informativ. Wird bei der textuellen Repräsentation eines Objekts jedoch beispielsweise ein Semantik tragendes Farbschema verwendet, so wären diese Objekte auch visuell informativ, da die Farben eine Information tragen und nur über das visuelle Wahrnehmungssystem interpretiert werden können.

Im weiteren Kontext dieser Arbeit ist die Bezeichnung 'visuell' in Sinne von rein visuell, bzw. visuell informativ zu sehen.

Der Autor in [Koh] fasst die Eigenschaften von visuellen Programmiersprachen folgendermaßen zusammen:

Visuelle Programmiersprachen sind Sprachen, die

- visuelle Elemente enthalten, die von syntaktischer und semantischer Bedeutung sind
- ihre Syntax aus der räumlichen Anordnung und den Verbindungen zwischen Grafiken und Texten erhalten
- ihre Semantik aus der Interpretation der syntaktischen Strukturen beziehen
- die semantische Anreicherung der Programmkonstrukte durch visuelle Attribute erhalten
- in ihrer Syntax weniger restriktiv sind bzgl. der Art und Weise wie ein Programm ausgedrückt werden kann

Nach [Koh] entsteht durch die oben genannten Eigenschaften visueller Programmiersprachen ein Freiraum zur Untersuchung und Entwicklung von Programmiermechanismen, die bisher unversucht waren.

2.2 Mentale Aspekte der Programmierung

Bei der visuellen Programmierung ist die Einfachheit und das Abgestimmtsein der Programmierumgebung auf den Benutzer und seine Denkweise der zentrale Usability-Faktor. Die allgemeinen Ziele der Forschung im Bereich visueller Programmiersprachen werden in [J.W] folgendermaßen zusammen gefasst:

1. die Programmierung für bestimmte Zielgruppen leichter verstehbar machen
2. die Korrektheit durchgeführter Programmieraufgaben verbessern
3. die Geschwindigkeit bei der Durchführung von Programmieraufgaben steigern

Um diese Ziele zu erreichen, nennt [J.W] die vier allgemeinen, in visuellen Programmiersprachen verwendeten Strategien:

1. **Greifbarkeit/Konkretheit**
   Gemeint ist das Gegenteil von Abstraktheit. Das bedeutet, dass ein Aspekt eines Programms durch Benutzung bestimmter Instanzen, die für den Benutzer greifbaren und verstehbaren Objekten entsprechen, ausgedrückt werden kann. Der Programmierer kann also beispielsweise einen semantischen Aspekt auf einem speziellen Objekt oder Wert ausdrücken.

2. **Unmittelbarkeit**
   'Das Gefühl haben, dass man direkt das Objekt bearbeitet.' Aus kognitiver Perspektive bedeutet die Unmittelbarkeit, dass die Anzahl auszuführender Aktionen auf dem Weg zu einem Ziel möglichst gering und der Weg dorthin möglichst kurz ist. Bei gegebener Konkretheit wäre ein Beispiel für Unmittelbarkeit, dass der Programmierer die Semantik eines Objekts oder Wertes direkt manipulieren kann, anstatt diese Semantik textuell zu beschreiben.

3. **Deutlichkeit**
   Ein Aspekt der Semantik ist deutlich, falls dieser direkt angegeben wird, ohne dass der Programmierer diesen ableiten muss. Ein Beispiel für Deutlichkeit ist, dass das System explizit Daten- oder Kontrollflüsse durch Kanten zwischen den betroffenen Variablen oder Anweisungen anzeigt.

4. **Direktes visuelles Feedback**
   Effekte der Programmbearbeitung sollen sofort angezeigt werden. Dazu wurden vier Level des Feedbacks eingeführt:

Level 1 Dem Computer wird keine Semantik impliziert, somit gibt es auch kein Feedback bzgl. des Programms an den Programmierer. Beipiel für Level 1 wäre ein Entity-Relationship-Diagramm zu Dokumentationszwecken.

Level 2 Der Programmierer kann Feedback über einen Teil des Programms erhalten, jedoch wird dieses Feedback nicht automatisch geliefert. Compiler unterstützen Level 2 Feedback minimal, Interpreter etwas mehr, da diese nicht auf endgültige Ausgabewerte begrenzt sind.

Level 3 Der Programmierer erhält automatisch stufenweises semantisches Feedback beim Bearbeiten des Programms und alle angezeigten Werte werden automatisch aktualisiert. Dadurch wird die Konsistenz von Anzeigestatus und Systemzustand sichergestellt. Die automatische Neuberechnung von Werten bei der Tabellenkalkulation entspricht diesem Level.

Level 4 Auf diesem Level reagiert das System auf Programmänderungen und andere Ereignisse wie Mausklicks etc. und stellt dabei sicher, dass alle dargestellten Daten genau den Zustand des Systems repräsentieren.

Möchte man in irgendeiner Form eine Programmierumgebung für Benutzer schaffen, so muss man sich zuerst die erwartete Benutzergruppe und deren Situation klar machen. Dabei gilt die Bezeichnung 'Benutzer' in diesem Fall sowohl für Anfänger als auch für Experten und man sollte sich die folgenden Fragen stellen: "Für wen ist diese Programmierumgebung?", "Was wollen die Benutzer damit tun?" und "Welche Aspekte der Programmieraktivität werden von der Programmierumgebung betroffen sein?" [T.R].

Neben den genannten, konkret auf die visuelle Programmierung abgestimmten Strategien, gibt es drei weitere sehr wichtige und psychologisch interessante Betrachtungsweisen im Kontext der mentalen Aspekte der Programmierung. Diese Aspekte sind bei der Entwicklung einer Programmierumgebung zu berücksichtigen und werden in [T.R] genannt.

Zum einen ist dies der Bereich der Psychologie des Programmierens, welcher die Bewegung zwischen Problem- und Programmdomäne betrachtet. Dieser Bereich bezieht sich auf die Bedeutung des Codes und beschäftigt sich aus Sicht des Programmierers mit den Fragestellungen "Wie löse ich das Problem?" (Problemdomäne) und "Was bedeutet dieser Code?" (Programmdomäne). Der zweite Bereich ist die Mensch-Computer-Interaktion (Human Computer Interaction, HCI), welches sich auf die Interaktion des Benutzers mit dem Notationssystem beschäftigt. Der dritte zu nennende Bereich ist eine sehr 'high-level' Sicht auf die Aktivität des Programmierens an sich und ist als 'Framework der kognitiven Dimensionen' zusammengefasst.

2.2.1 Psychologie des Programmierens

In [T.R] wird zunächst einmal festgestellt, dass Daten in einer benutzbaren Form repräsentiert werden müssen, damit sie überhaupt zu Informationen werden. Die Wahl der Datenrepräsentation in einer Programmierumgebung beeinflusst die Usability der Programmierumgebung. Wie gut oder schlecht eine Repräsentation ist, hängt jedoch von Zweck und Kontext ab und

kann nicht allgemein gültig bestimmt werden. Die Psychologie des Programmierens fasst jedoch grundlegende Erkenntnisse bezüglich der Repräsentation von Daten in Programmierumgebungen in zwei Maximen zusammen:

1. Jede Notation hebt bestimmte Arten von Information hervor, jedoch auf Kosten anderer Informationen.
   - Nicht alles kann gleichzeitig hervorgehoben werden.
   - Hebt eine Sprache den Datenfluss hervor, so kann es gut sein, dass dadurch die Darstellung des Kontrollflusses vernachlässigt wird.
   - Teil des Entwurfs ist es also, die vernachlässigte Information sichtbarer zu machen.
2. Beim Suchen nach Informationen muss es einen kognitiven Zusammenhang zwischen der mentalen Repräsentation (innere Vorstellung) und der externen Repräsentation (Darstellung durch das Programm) geben.
   - Wenn die mentale Repräsentation in Form eines Kontrollflusses vorliegt, so wird für den Benutzer eine Datenfluss-Repräsentation schwer zu verstehen sein.

Betrachtet man diese beiden Maximen, so bedeutet das, dass ein Programmiersystem, bzw. eine Programmierumgebung nur dann Erfolg haben wird, wenn die dem Programmierer zur Verfügung gestellten Sprachmittel für die vom Programmierer auszuführenden Aufgaben geeignet sind. Ferner müssen die zur Verfügung stehenden Sprachkonstrukte der mentalen Repräsentation des Programmierers entsprechen.

Hinsichtlich dieser beiden genannten Maximen werden in [T.R] die folgenden Aspekte für die Entwicklung von Programmierumgebungen näher diskutiert:

- **Mentale Repräsentation**
  Die mentale Repräsentation eines Programms liegt auf einer höheren Ebene als der des reinen Codes. Die Dislokation und Zerstreuung mental zusammengehöriger Konstrukte wurde als Hauptproblem beim Erlernen des Programmierens erkannt. Studien von Spohrer und Soloway [SS86] zufolge entstehen bei Anfängern die meisten Fehler aufgrund einer falschen Vorstellung über die Sprachkonstrukte.
- **Reihenfolge der Programmentwicklung**
  Die Entwicklung eines Programms erfolgt nicht linear. Programmierer schreiben ihren Programmtext nicht von oben nach unten herunter, noch arbeiten sie top-down vom höchsten mentalen Konstrukt zum kleinsten. Daher sollte auch die Reihenfolge der Programmbearbeitung bzw. -entwicklung allein beim Programmierer gelassen und nicht von der Programmierumgebung vorgegeben werden.

- **Effekt der Umgebung**
  Wenn Programmierer in einer textbasierten Programmiersprache entwickeln, dann entwickeln sie ihren Code in Abschnitten, die mentalen Blöcken oder Schemata entsprechen und die in den Kontext des bisher vorhandenen Programms eingefügt werden. Der Programmierer muss für die Fortführung seiner Gedankengänge das bisher erstellte Programm vor Augen haben und verstehen können. Der Autor in [Dav] stellt diesbezüglich eine direkte Verbindung zu der Umgebung der Programmierung her: Ein Editor, der einen einfachen Zugang zu einem großen Bereich an Code ermöglicht, macht den Entwicklungszyklus erheblich einfacher. Das so genannte 'window of access' muss groß genug sein, um ein gute Übersichtlichkeit für den Programmierer zu gewährleisten.

- **Vom Problem zum Programm**
  Der Programmierprozess kann verstanden werden als eine Abbildung von Problemen aus der Problemdomäne in Konstrukte der Programmdomäne. Für den Programmierer ist es daher schwer, mit Entitäten der Programmdomäne umzugehen, die keine Entsprechungen in seiner Problemdomäne haben. Aufgabenspezifische Sprachen haben per Definition einen großen Anteil an Entitäten, die direkt auf die Problemdomäne abbilden. Visuelle Programmiersprachen sind nicht die einzige Möglichkeit, aufgabenspezifische Programmiersprachen zu erstellen, aber sie können sehr effektiv sein.

- **Verstehen und Evaluieren des Programms**
  Programmierer müssen während ihres Entwicklungszyklus die Möglichkeit haben, sowohl vollständige als auch unvollständige Programme zu lesen und zu evaluieren. Gerade für Anfänger ist es wichtig, einen Programmteil zu testen, bevor er dem gesamten Programm-Code hinzu gefügt wird. Aber auch Experten debuggen ihre Programme und unterteilen sie dazu oft in kleine zu untersuchende Abschnitte. Dieses Vorgehen muss von der Programmierumgebung unterstützt werden.

Anhand der genannten Aspekte wird in [T.R] geschlussfolgert, dass die Verständlichkeit einer Programmiersprache von der Übereinstimmung zwischen der Art und Weise ihrer Strukturierung und der Art der damit zu formulierenden Probleme abhängt.

2.2.2 Human Computer Interaction (HCI)

Wie bereits erwähnt, beschäftigt sich die Psychologie des Programmierens hauptsächlich mit den Aspekten der Programmierung, die die Bewegung zwischen Problem- und Programmdomäne betreffen. Die HCI hingegen konzentriert sich mehr auf die interaktiven Situationen der Programmierung, als auf den Entwurf von Notationen.

Zwei zentrale Aspekte der HCI, die sich auf die Interaktion mit dem Programm-Code beziehen, sind in [T.R] genannt. So ist es für den Entwurf einer Programmierumgebung wichtig,

dem Benutzer eine gute Bedienung des Layouts bzw. des Textes zu ermöglichen. Auch ist es wohl bekannt, dass Programmierer sich innerhalb ihres Programm-Codes beliebig bewegen und Suchoperationen ausführen. Dieses Verhalten sollte also durch die Programmierumgebung unterstützt werden und dem Programmierer eine leichte Vor- und Zurück-Navigation in seinem Programm-Code ermöglichen.

2.2.3 Das Framework der kognitiven Dimensionen

Neben den beiden zuvor betrachteten Bereichen der mentalen Aspekte, welche aus verschiedenen Gesichtspunkten allgemeine mentale Grundsätze für die Entwicklung von Programmierumgebungen betrachten, gibt es noch das in [T.R] beschriebene Framework der kognitiven Dimensionen. Dieses ist eine Evaluationstechnik für Systeme der visuellen Programmierung und definiert Begriffe, um die kognitiv-relevanten Aspekte und deren Abgrenzung zueinander zu erfassen. Es umfasst die unterschiedlichen Aktivitäten und Situationen des visuellen Programmierprozesses.

Die grundlegende Frage beim Framework der kognitiven Dimensionen besteht darin, ob die vom Benutzer gewollten Aktivitäten durch die Software angemessen unterstützt werden. Somit besteht die Evaluierung darin, die gewollten Aktivitäten zu klassifizieren, die kognitiven Dimensionen zu analysieren und zu entscheiden, ob die Anforderungen der Aktivitäten getroffen sind oder nicht.

In [T.R] werden dazu die folgenden kognitiven Dimensionen im Bezug auf die Programmier-Aktivität genannt:

- **Abstraktionsgrad**
  Diese Dimension beschäftigt sich mit der Frage, welches der minimale und maximale Grad von Abstraktion in der Programmiersprache ist. Programmiersprachen können eingeteilt werden in abstraction-hating, abstraction-tolerant (C und HyperTalk) oder abtraction-hungry (Smalltalk), basierend auf ihrem minimalen Startlevel der Abstraktionen und ihrer Bereitschaft oder Forderung weitere Abstraktionen zu akzeptieren. Abstraction-hungry Systeme haben offensichtliche Nachteile: Jedes Programm erfordert eine neue Abstraktion. Der Programmieranfänger kann jedoch nicht einfach Problementitäten in Domänenentitäten abbilden. Sie brauchen notwendigerweise eine relativ lange Zeit zur Einrichtung, da die entsprechenden Abstraktionen erst definiert werden müssen, bevor die eigentlichen Ziele des Programmierers angegangen werden können. Die meisten End-Benutzer wehren sich vor abstraction-hungry Systemen. Insbesondere in Bereichen, in denen Flexibilität ein wichtiges Kriterium ist, eignen sich abstraction-hungry Systeme gut, da sie die Einführung weiterer Abstraktionen ermöglichen. Gut gewählte Abstraktionen können auch den Schutz vor Fehleranfälligkeit erhöhen, da durch Deklaration der Identifier Schreibfehler zur Compile-Zeit und nicht erst zur Laufzeit entdeckt werden können.

- **Nähe der Abbildung**
  Hier geht es darum, welche programmiertechnischen Konzepte vom Benutzer erlernt werden müssen. Wie in der Psychologie bereits erwähnt, erfordert Programmierung eine Abbildung zwischen einer Problemwelt und einer Programmwelt. Je näher die Programmierwelt an der Problemwelt liegt, desto einfacher sollte es sein, ein Problem zu lösen. Idealerweise sollten die Problem-Entitäten im Aufgabenbereich des Benutzers direkt auf aufgabenspezifische Programm-Entitäten abgebildet werden können.

- **Konsistenz**
  Die Konsistenz einer Programmiersprache besteht darin, dass, wenn der Benutzer einen Teil der Sprache erlernt hat, er daraus weitere Konzepte der Programmiersprache ableiten kann. Eine konsistente Sprache kann leichter erlernt werden. Es gibt dabei zwei Bedeutungen der Konsistenz:

  1. Eine Sprache kann mehr oder weniger zufällig zusammengefügt werden, und zwar auf eine beliebige Art und Weise
  2. Eine Sprache kann von ihrem Entwickler als hochgrading konsistent angesehen werden, kann jedoch bei einem Lernenden Probleme hervorrufen, da sich dessen Verständnis der Struktur von dem des Entwicklers unterscheidet, also inkonsistent ist zum Verständnis des Entwicklers.

- **Komplexität der visuellen Elemente**
  Wie viele Symbole oder grafische Entitäten werden benötigt, um eine Begrifflichkeit auszudrücken? Einige Notationen brauchen viele Symbole oder viel Platz, um bestimmte Ziele zu erreichen, die andere Notationen auf kompaktere Weise darstellen können. Notationen, die eine sehr direkte Abbildung zur Problemdomäne haben, werden zur Zielerreichung weniger Lexeme benötigen und werden dafür in ihren Möglichkeiten eher eingeschränkt erscheinen. Beinhaltet ein visuelles Konstrukt übermäßige Komplexität, so sind die kognitiven Auswirkungen offensichtlich: je mehr Material vom Benutzer zum Verstehen des Konstrukts gescannt werden muss, desto weniger dieser Konstrukte können im 'Arbeits-Gedächtnis' des Benutzers gehalten werden und desto häufiger muss der Programmierer durch das Programm suchen und zwischen den vorhandenen Konstrukten navigieren, um sich deren Funktionen ins Gedächtnis zu rufen. Ist das Abstraktionsniveau eines visuellen Konstrukts jedoch zu gering, so unterscheiden sich gänzlich unterschiedliche Programme visuell kaum, und es wird für den Benutzer schwer, diese zu 'scannen', um die Unterschiede zu erkennen.

- **Fehleranfälligkeit**
  Die gewählte Notation sollte Flüchtigkeitsfehler vermeiden.

- **Schwer verständliche Operationen**
  Gibt es Stellen, an denen der Benutzer zu Hilfsmitteln wie beispielsweise schriftlichen Notizen greifen muss, um zu verstehen, was gerade passiert? Programmiersprachen sollten so genannte 'brain-twisters' vermeiden.

- **Versteckte Abhängigkeiten**
  Ein versteckte Abhängigkeit ist eine Abhängigkeit zwischen zwei Objekten in der Art, dass eine Komponente von der anderen abhängig ist, diese Abhängigkeit jedoch nicht offensichtlich erkennbar ist. Man sollte sich beim Entwurf einer Notation überlegen, ob jede Abhängigkeit in beide Richtungen gut visualisiert wird. Beispiele für versteckte Anhängigkeiten: Zellen in Tabellenverarbeitungsprogrammen, Style-Definitionen in Word, HTML-Links.

- **Frühzeitige Festlegung**
  Manchmal ist der Benutzer gezwungen eine Entscheidung zu treffen, bevor entsprechende Informationen verfügbar sind. Dieses Problem tritt dann auf, wenn
  - die Notation viele interne Abhängigkeiten enthält
  - das Medium oder die Arbeitsumgebung Contraints über die Reihenfolge der Arbeitsschritte enthält
  - diese Reihenfolge unangebracht ist

- **Schrittweise Auswertung**
  Dieser Punkt entspricht dem in der Psychologie des Programmierens erwähnten Aspekt 'Verstehen und Evaluieren des Programms'. Hierbei kommt es darauf an, ob der Programmierer nur teilweise fertige Programme ausführen kann, um ein Feedback zu einer bestimmten Aktion zu bekommen. Gerade Anfänger benötigen die Möglichkeit, ihren eigenen Problemlösungsprozess in häufigen Intervallen bewerten zu können.

- **Sekundäre Notation**
  Die sekundäre Notation beschreibt Layout, Farben oder andere Hinweise die Programmierer nutzen können, um zusätzliche Bedeutungen über die offizielle Semantik der Sprache hinaus auszudrücken. Viele Programmiersprachen ermöglichen es, zusätzliche Informationen durch andere Mittel als ihre formale Syntax auszudrücken. Beispielsweise durch Einrückung, Kommentare, Namenskonventionen, Programmierkonstrukte und Gruppierung zusammengehöriger Anweisungen.

- **Aussagekräftigkeit von Rollen**
  Der Programmierer sollte erkennen können, wie jede einzelne Komponente eines Programms in Beziehung zum Ganzen steht. Notationen mit aussagekräftigen Rollen machen es leichter zu erkennen, warum der Programmierer die Struktur auf eine bestimmte Art und Weise gebaut hat. Die Aussagekraft kann vermutlich durch die Benutzung aussagekräftiger Variablennamen und sekundärer Notationen, gut strukturierter Modularität verbessert werden. Die Aussagekraft von Rollen kann durch eine explizite Ebene für Beschreibungen verbessert werden.

- **Viskosität**
  Wie viel Aufwand ist notwendig, um eine einzelne Änderung zu machen? Ein viskoses System benötigt viele Benutzeraktionen, um zu einem gewünschten Ziel zu gelangen.

Ein viskoses Textverarbeitungssystem wäre beispielsweise, wenn zur Änderung des Schriftstils einer Überschrift alle vorhandenen Überschriften einmal angefasst werden müssten. Man unterscheidet zwischen

  - Wiederholungs-Viskosität: viele gleichartige Aktionen
  - Knock-out Viskosität: weitere Aktionen sind notwendig, um die Konsistenz wieder herzustellen

- **Sichtbarkeit**
  Ist jeder Bereich des Codes gleichzeitig sichtbar, oder ist es zumindest möglich, beliebige zwei Teile nebeneinander zu stellen? Wenn der Code aufgeteilt ist, ist es zumindest möglich zu wissen, in welcher Reihenfolge er zu lesen ist? Systeme, die Informationen in Verkapselungen verbergen, reduzieren die Sichtbarkeit.

Diese kognitiven Dimensionen liefern grundlegende Aspekte für die Evaluierung visueller Systeme, was im weiteren Verlauf der Arbeit von Interesse sein wird.

2.3 Vergleich textueller und visueller Programmiersprachen

In den Abschnitten der Psychologie des Programmierens und der HCI wurden allgemeine Aspekte der Programmierung (textuell oder visuell) betrachtet. Das Framework der kognitiven Dimensionen hat weitere kognitive Dimensionen zur Evaluierung von visuellen Systemen hinzugefügt. Unter Betrachtung der in diesen Abschnitten diskutierten Aspekte sind in Tabelle 2.1 die zentralen Eigenschaften textueller und verbaler Notationen aufgeführt und gegenüber gestellt [Koh].

Merkmal	visuell	verbal
Eigenschaften	• Visuelle Elemente sind **syntaktisch und/oder semantisch bedeutungstragend**. • **Syntax** ergibt sich aus verwendeten Graphiken und Texten, ihrer **räumlichen Anordnung** und ihrer **Verbindungen**. • **Semantik** beruht auf **Interpretation der syntaktischen Struktur** und der semantischen Anreicherung syntaktisch äquivalenter Programmkonstrukte durch visuelle Attribute. • **mehrdimensionale Darstellung**: es können verschiedenste anwendungsspezifischen Zeichen, Symbole und Bilder genutzt werden.	• **Visuelle Elemente** sind syntaktisch und semantisch **bedeutungslos**. • Syntax ergibt sich aus **linearer Anordnung** der lexikalischen Elemente. • Semantik beruht auf der **Interpretation** jener Programmkonstrukte, die bei **syntaktischer Analyse** erkannt werden. • **Eindimensionaler Zeichenstrom**, es gibt nur die Nachfolgebeziehung zwischen den elementaren Bestandteilen (Buchstaben, Tokens) der Repräsentation. Die zweidimensionale Formatierung ist zumindest für den Übersetzer/Interpreter der Programmiersprache ohne semantische Bedeutung. • Es wird hauptsächlich die **linke Gehirnhälfte** des Menschen beansprucht, welche vorwiegend für das Verarbeiten geschriebener und gesprochener Informationen sowie logisches und analytisches Denken verantwortlich ist. Die **Informationsverarbeitung** erfolgt dabei **sequentiell**.
Ziele	• Vereinfachung der Programmierung. • Methodik und somit Korrektheit bei Programmieraufgaben verbessern. • Geschwindigkeit für Problemlösung steigern.	• Textuelle, strukturierte Beschreibung von Problemen.

Merkmal	visuell	verbal
Vorteile	• Bilder werden schneller verarbeitet und hinterlassen einen stärkeren Eindruck. • Beanspruchen die **rechte Gehirnhälfte**, welche für die Verarbeitung bildlicher und räumlicher Informationen sowie ästhetischer gefühlsbezogener Kognition zuständig ist. Hier erfolgt meist eine **parallele Informationsverarbeitung**, dadurch können mehr Informationen gleichzeitig wahrgenommen werden. • Visuelle Darstellung quantitativer Attribute • Mehrdimensionale Darstellung • Leichtere Erlernbarkeit durch metamophorische Bedeutungen • Bei Bildern sind keine Sprachbarrieren zu überwinden. • Kein tiefes Verständnis von computerspezifischen Methoden notwendig • Querbezüge in der Grafik direkt durch Querverbindungen darstellbar • Extrem hohes Abstraktionsniveau • Strukturelles Editieren: Spezifikation von syntaktisch inkorrekten Programmen nicht möglich • Gesteigerte Entwicklungsgeschwindigkeit • Benutzung von Metaphern einer bestimmten Domäne sehr vorteilhaft • Durch Layoutfreiheiten bessere Darstellung von bestimmten Strukturen und zusätzlichen informellen Zusatzinformationen	• Einfache/bekannte Implementierungstechnik • Einheitliche Terminologie vorhanden • Eindeutige Regeln, wie Programme zu lesen sind • Kompakte Darstellung, dadurch wesentlich mehr textuelle als grafische Information auf einer Seite • Menschen sind darauf trainiert, Texte zu lesen und zu schreiben. • Es existiert eine Fülle verschiedener Abstraktionsmechanismen (Prozeduren, Module, Klassen mit Vererbung). • Beim Ausdrucken von Programmen ist man zumindest auf die üblichen zwei Dimensionen beschränkt.

Merkmal	visuell	verbal
Nachteile	• Aufwändige Implementierung • Keine einheitliche Terminologie • Manche Begriffe sind nur schwer visualisierbar, komplexe Sachverhalte sind oft schwer mit visuellen Formen darzustellen. • Mögliche Fehlinterpretationen der ursprünglich gewählten Methaphern bei Wiederverwendung in unterschiedlichen Kontexten • Oft existieren keine einfachen Bilder für zu abstrakte Konzepte. • Benutzung von Bildern kann zu Missverständnissen führen. • Unterschiedliche Betrachter interpretieren Bilder häufig anders. • Die Leserichtung bei komplizierten Sachverhalten ist meist unklar. • Es gibt keine Regeln, wie Diagramme zu lesen sind. • Die Wartbarkeit hängt stark vom Verstehen der Programme ab. • Für jede Sprache wird ein eigener Spezialeditor benötigt.	• Eindimensionale Darstellung • Verbale Informationen werden sequentiell verarbeitet. • Die verwendete Programmiersprache und deren Verwendung muss erlernt werden. • Benutzung entspricht meist nicht ohne Weiteres der intuitiven Denkweise von Benutzern. • Keine Querbezüge darstellbar. • Aspekte/Entitäten der Anwendungsdomäne müssen oft erst selbst modelliert werden.
kognitive Aspekte	• Inkrementelle Entwicklung bietet mehr Möglichkeit für visuelles oder räumliches Denken. • Sehr effektiv bzgl. dem in 2.2.3 genannten Ziel der 'Nähe der Abbildung' • Schlecht gestaltete Grafiken stiften auf Ebene der sekundären Notation mehr Verwirrung als schlecht strukturierte Texte. • Hoher Platzbedarf und Aufwand bei der Konstruktion, daher weniger Informationen gleichzeitig darstellbar • Editor nur intuitiv bei wenigen Sprachkonstrukten. • Der Enthusiasmus schlägt rasch in Frustration um, wenn geeignete Sprachkonzepte zur Bewältigung komplexer Problemstellungen fehlen oder umständliche Interaktionsmechanismen das schnelle Editieren größerer Programme verhindern. • Skalierungsproblem	• Können Wahrnehmungshinweise wie die Einrückung, die Wahl der Schriftart oder symbolische Hinweise enthalten • Meist flexibler und kompakter zur Bewältigung komplexer Problemstellungen • Nur eindeutige Interpretation möglich • Sind weit entfernt vom Ziel der 'Nähe der Abbildung' (siehe 2.2.3) • Enthalten viele syntaktische Entwurfsfeatures, die dazu führen, dass Flüchtigkeitsfehler auftreten oder diese dann auch schwer zu finden sind. • Versteckte Abhängigkeiten (Nebeneffekte) möglich. • (Gerade von Anfängern) nicht eingehaltene Coding-Conventions führen zu schlechter Lesbarkeit (Wartbarkeit). • Kein einfacher Drag-and-Drop Mechanismus von Elementen möglich • Formulierung oft nicht intuitiv

Tabelle 2.1: Gegenüberstellung visueller und verbaler Programmiersprachen

Aus Tabelle 2.1 lässt sich bezüglich visueller Systeme zunächst herleiten, dass Bilder nur eine Ergänzung zu verbaler Verständigung sein können. Nicht alles ist bildlich darstellbar, was verbal beschreibbar ist. Im formalen Bereich ist Grafik allein kaum jemals ausdrucksstärker als Text. Die Darstellung von Programm-Code in grafischer Form ist oftmals eher verständnishemmend als verständisfördernd. Eine gute Gestaltung, der Einsatz von sekundärer Notation und Mechanismen, um relevante Daten zu zeigen, machen eine gute visuelle Sprache aus.

Die textuelle Beschreibung von Problemen erfordert das Erlernen der Zielsprache (Syntax) sowie das Verstehen der Konzepte (Semantik). Das Verstehen einer Programmiersprache hängt ab von der Übereinstimmung zwischen der Art und Weise, wie sie strukturiert ist, und wofür sie eingesetzt werden soll. Eine Möglichkeit, die kognitive Eignung textueller Beschreibungen zu verbessern besteht darin, Hinweise einzufügen, die die Zugänglichkeit der Informationen verbessern.

2.4 Konzepte visueller Programmiersprachen

Im Bereich der visuellen Programmierung gibt es im Gegensatz zur textuellen Notation verschiedene Möglichkeiten, Daten bzw. Programme zu repräsentieren. Die folgenden Tabellen sollen anhand von [Koh] einen Überblick der verschiedenen visuellen Darstellungskonzepte und deren Eigenschaften und Einsatzbereiche liefern.

2.4.1 Steuerflussorientierte visuelle Programmiersprachen

Sprachkonzept: Befehlsaktivierung durch die Programmsteuerung

Programmierparadigma: imperativ

Visualisierung:

- Anweisungssequenzen
- Komponentennetze
- Transitionsnetze

Eigenschaften:

Anweisungssequenzen:

- Operationen werden in der Reihenfolge, in der sie aufgeschrieben sind, ausgeführt
- Entsprechen sehr stark den textuellen Sprachen

Komponentennetze:

- Informationsverarbeitung erfolgt in den Softwarekomponenten
- Hohes Abstraktionsniveau
- Sehr flexibel

Transitionsnetze:

- Beschreiben Automaten und Petrinetze (reaktive Systeme)
- Exakte Spezifikation von Systemeigenschaften
- Beweisbare Aussagen über das Systemverhalten

Anwendungsgebiete:

- Transitionsnetze: mathematische Modelle, meist reaktive Systeme.
- Ablaufdiagramm: Veranschaulichung der Funktionsweise eines Computerprogramms

- Nassi-Shneiderman-Diagramme: Programmierparadigma der Strukturierten Programmierung

kognitive Aspekte:

- Puzzlediagramme: syntaktische Fehler können teilweise direkt ausgeschlossen werden
- Hohes Abstraktionsniveau
- Sehr flexibel
- Drag and Drop ist eine weitere Motivation, für ungeübte Benutzer

2.4.2 Datenflussorientierte visuelle Programmiersprachen

Sprachkonzept: Verfügbarkeit von Daten bestimmt die Ausführungsreihenfolge

Programmierparadigma: imperativ

Visualisierung:

- Datenflussdiagramme
- Verschiedene Notationen
- UML: Aktivitätsdiagramm

Eigenschaften:

- Stellt die Art der Verwendung, die Bereitstellung und Veränderung von Daten innerhalb eines Programms dar
- Weit verbreitet, auch kommerziell erfolgreich
- Parallele Ausführung datenunabhängiger Operationen
- Keine Variablen
- Nebenwirkungsfrei (Referenzielle Transparenz)
- Verzweigungen, Schleifen und Fallunterscheidungen schwer zu realisieren
- Daten nicht passiv, sondern aktivieren die Operationen
- Operation wird ausgeführt, wenn alle Eingabedaten zur Verfügung stehen, daher kein Befehlszähler nötig

Anwendungsgebiete:

- Wesentliches Modellierungsinstrument der strukturierten Analyse
- Geeignet für Pipeline-Programmierung
- Visualisierung der Programmausführung
- Wiedergabe des Datenflusses eines Prozesses oder einer Tätigkeit

kognitive Aspekte:

- Leiden nicht so stark unter dem Spaghetti-Effekt, denn es werden Funktionen anstelle von Objekten miteinander verbunden.
- Variablen sind konzeptionell unnötig, Wegfall von Fehlermöglichkeiten
- Keine Querbeziehungen, daher leichtes Top-Down-Modellieren
- Hohe Viskosität bei schlechten Diagramm-Editoren
- Lässt Kontrollfluss des Programms außer Acht
- Implizite Ausführungsreihenfolge
- Nebeneffektfreiheit bewirkt starke Einschränkung der Möglichkeiten
- Verzweigungen, Fallunterscheidungen und Schleifen schwer zu realisieren, daher schwer verständliche Diagramme möglich

2.4.3 Funktionsorientierte visuelle Programmiersprachen

Sprachkonzept: wie datenflussorientierte VP-Systeme, jedoch um klassische funktionale Konzepte erweitert

Programmierparadigma: deklarativ

Visualisierung: Funktionendiagramm

Eigenschaften:

- Funktionen höherer Ordnung
- Polymorphe Funktionen
- Implizite Typisierung
- Es gibt keine Variablen und Wertzuweisungen.
- Nutzen die Vorteile des funktionalen Paradigmas

Anwendungsgebiete: mathematische Formalisierung von Problemen

kognitive Aspekte:

- Typisierung: Vermeidung/schnelle Auffindbarkeit von Flüchtigkeits- oder Typfehlern
- Es gibt kaum funktionsorientierte visuelle Programmiersprachen, da die funktionale Programmierung aus dem Bedürfnis nach mathematischer Formalisierung entstand und textuelle Notationen dafür bedeutend besser geeignet sind als grafische Darstellungen.

2.4.4 Objektorientierte visuelle Programmiersprachen

Sprachkonzept: Visuelle Darstellung der Klassenstruktur

Programmierparadigma: objektorientiert

Visualisierung: Diagramme, wobei Objekte Schnittstellen zur Kommunikation mit anderen Objekten haben

Eigenschaften:

- Aufteilung der Software-Erstellung auf zwei Ebenen:
  1. Erstellung von Programmbausteinen mittels verbaler Programmiersprachen und Klassenbibliotheken
  2. Erstellung von Applikationen durch visuelle Verbindung von Bausteinen zu komplexen Einheiten
- Kommunikation der Bausteine mittels Attributen, Ereignissen, Nachrichten
- Elementare Funktionen müssen im Gegensatz zu anderen Konzepten visueller Programmiersprachen allerdings immer noch textuell erstellt werden.

Anwendungsgebiete: Visuelle Darstellung der Klassenstruktur

kognitive Aspekte:

- Enge Verbindung von Problem und Lösung legt eine visuelle Repräsentation mit den Möglichkeiten direkter Manipulation nahe
- Tendenz zur Unübersichtlichkeit
- Probleme mit Objekterzeugung und Polymorphismus

2.4.5 Constraintorientierte visuelle Programmiersprachen

Sprachkonzept: Programme bestehen aus Objekten und Constraints zwischen den Objekten

Programmierparadigma: logisch-deklarativ

Visualisierung: Anordnung und Verbindung von Piktogrammen/Icons

Eigenschaften:

- Der Programmablauf erfolgt durch automatischen Mechanismus (Constraint-Satisfier), der dafür sorgt, dass die Constraints erfüllt bleiben, falls sich Objektattribute ändern.
- Keine algorithmischen Programmkonstrukte wie Zuweisungen, Prozeduren oder Ablaufstrukturen
- Zwischen Eingabe und Ausgabe wird nicht unterschieden
- Über-/Unterspezifikation führt zu keinen/zu vielen Lösungen
- Zu große Laufzeiten, selbst bei geringen Problemstellungen

Anwendungsgebiete: Simulation von Problemen, bei denen bestimmte Bedingungen zwischen Objekten gelten müssen

kognitive Aspekte:

- Die Idee, Probleme nur zu spezifizieren, aber nicht algorithmisch zu lösen, mag zwar verlockend erscheinen, die deklarative Programmierung mit Constraints stößt jedoch schnell an ihre Grenzen.
- Starke Begrenzung der möglichen Anwendung

2.4.6 Regelorientierte visuelle Programmiersprachen

Sprachkonzept: Programme werden als Folge von Regeln oder Bildtransformationen beschrieben

Programmierparadigma: deklarativ

Visualisierung: Anordnung und Verbindung von Piktogrammen/Icons

Eigenschaften:

- Ein Programm besteht aus einer Menge von Objekten, zugeordneter, grafischer Transformationsregeln mit optionalen Bedingungen und Aktionen.

- Eine Transformationsregel besteht aus einer linken (Muster) und einer rechten Grafik (Resultat).
- Durch kontinuierlichen Mustervergleich der umzuformenden Grafik mit den linken Seiten der Transformationsregeln ersetzt das VP-System die gefundenen Muster der Ausgangsgrafik durch die zugehörigen Resultatdarstellungen.
- Ein Programm wird durch eine Ausgangsgrafik gestartet und endet, wenn kein Ausschnitt mehr gefunden wird, der zu einer Regel passt.
- Konflikte durch mögliche Anwendung mehrerer Regeln (Lösung: Regelwahrscheinlichkeiten)
- Kombinatorische Explosion der Regelanzahl bei komplexeren Aufgabenstellungen

Anwendungsgebiete: einfache Simulationen für Chemie, Biologie, Elektronik

kognitive Aspekte:

- Hebt Bedingungen hervor, unter denen Aktionen ausgeführt werden
- Vernachlässigt evtl. die sequentielle Abfolge von Aktionen

2.4.7 Beispielorientierte visuelle Programmiersprachen

Sprachkonzept: Benutzer führt vor, System beobachtet

Visualisierung: Aufzeichnung von Benutzerinteraktion

Eigenschaften:

- Beobachten die einmal vom Benutzer vorgezeigte Lösung, ziehen Schlussfolgerungen und ein generieren Programm, welches die gleichen oder ähnliche Aufgaben lösen kann.
- Es wird nicht versucht, die Absicht des Benutzers zu erkennen, sondern vielmehr die am Bildschirm aufgezeichneten Aktionen in Programm-Code zu übersetzen.
- Schleifen und Verzweigungen sind vom System nur schwer zu erkennen.

Anwendungsgebiete: Beispielorientierte Systeme sind mit Ausnahme von Makrorekordern praktisch nicht anwendbar.

kognitive Aspekte:

- Besitzen nur geringe Flexibilität, da bereits kleine Abweichungen vom ursprünglichen Szenario zu Fehlfunktionen des Programms führen

- Die Ausdrucksmöglichkeiten des Benutzers sind auf visuelle Interaktionen beschränkt. Verzweigungen oder Schleifen in den Aktionen des Benutzers sind nur schwer erkennbar. Daher gibt es eine Diskrepanz zwischen den Interaktionen des Benutzers und dem generierten Code, die nur schwer aufzuheben ist.
- Unzureichende Ausdrucksstärke der visuellen Interaktion
- Semantische Lücke zwischen Benutzerinteraktion und generiertem Code

2.4.8 Formularorientierte visuelle Programmiersprachen

Sprachkonzept: Programme werden durch Ausfüllen von Programmblättern oder Formularen beschrieben

Visualisierung: Formulare

Eigenschaften:

- Einzelne Zellen der Formulare enthalten Daten oder Formeln zur Berechnung von Daten.
- Automatische Nachberechnung von Daten, die von anderen Daten abhängig sind
- Keine Unterstützung sequentieller Abläufe
- Keine algorithmischen Konstrukte
- Mangelnde Konzepte zur Datenabstraktion

Anwendungsgebiete: Für DB-Systeme bewährt, Beispiel: QBE (query by example)

kognitive Aspekte:

- Anfällig für versteckte Abhängigkeiten
- Schlechte Sichtbarkeit: haben zwei Ebenen (Daten und Formeln)

2.4.9 Multiparadigmenorientierte visuelle Programmiersprachen

Sprachkonzept: Kombinieren mehrere Konzepte der visuellen Programmierung

Visualisierung: Kombination verschiedener visueller Konzepte

Eigenschaften:

- Nahtlose Integration der unterstützten Modelle, ohne einen konzeptionellen oder technischen Bruch bei der Implementierung zu verursachen

- Der Anwender kann ein System mit den verschiedenen Paradigmen entwerfen, ohne selbst die Schnittstellen zwischen denselben implementieren zu müssen.
- Vorteile der verschiedenen Konzepte können genutzt werden.
- Gefahr schwer verständlicher Programme
- Notwendigkeit zu Kompromissen zwischen den Stärken der einzelnen Konzepte

Anwendungsgebiete: Vereinigung steuerfluss-, datenfluss- und objektorientierter Konzepte

kognitive Aspekte:

- Freie Wahl des Programmierers, welches Denkmodell zur Lösung einer Teilaufgabe herangezogen wird.
- Durch die große Komplexität ergeben sich Probleme
- Tendenz zu konzeptioneller Überladung. Der Programmierer ist somit nicht mehr in der Lage, aus den verschiedenen angebotenen Konzepten auszuwählen.

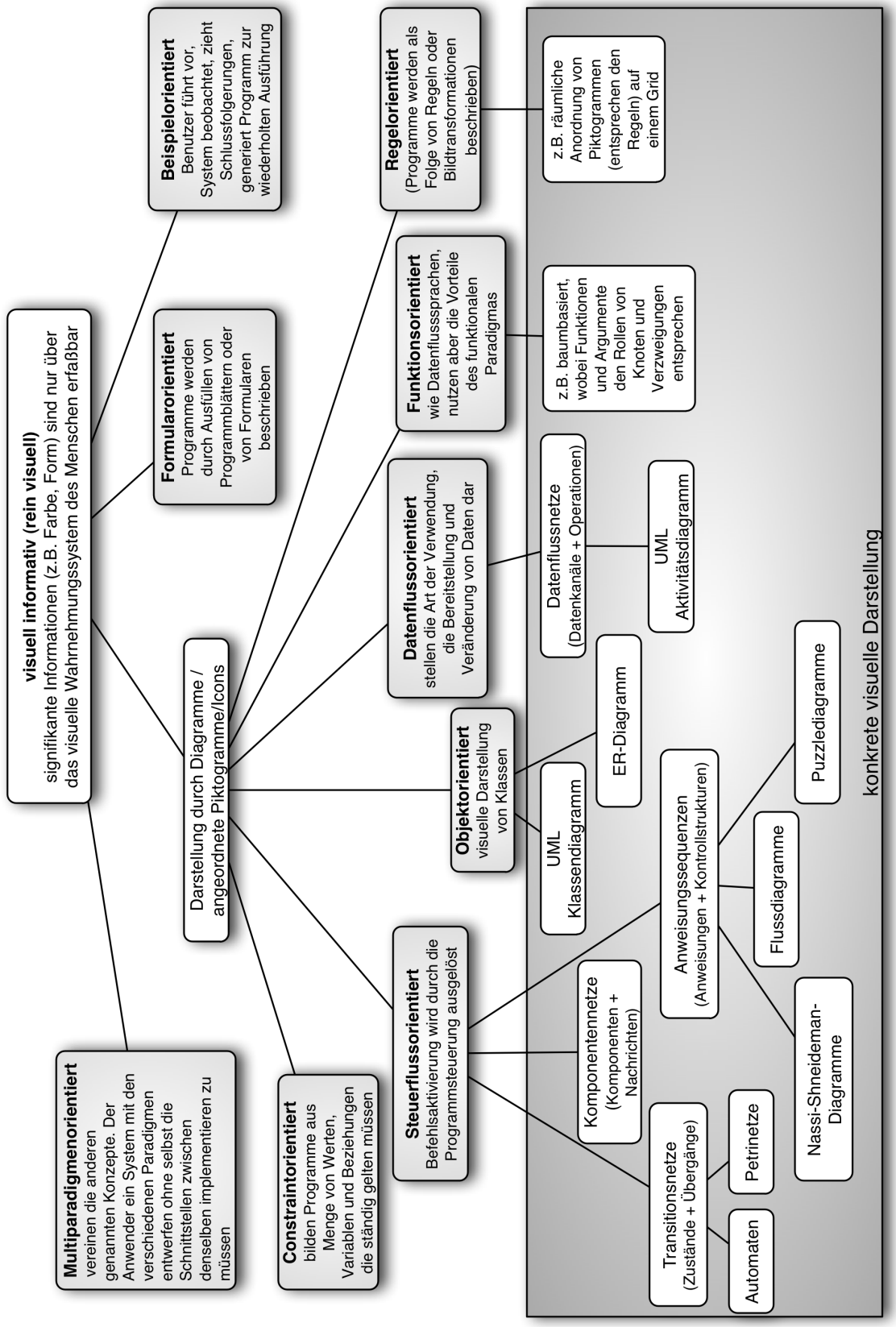


Abbildung 2.2: Übersicht visueller Programmierkonzepte

3 Evaluierung bestehender Systeme

Nachdem die Grundlagen und verschiedenen Prinzipien der visuellen Programmierung für End-Benutzer erörtert wurden, soll nun betrachtet werden, welche Produkte und Methoden es bereits für die Domäne der Prozessmodellierung in Webportalen gibt. Dabei sollen diese Produkte bezüglich ihrer Gebrauchstauglichkeit für die genannte Domäne evaluiert werden.

Der Prozess der Analyse und Evaluierung vorhandener Softwareprodukte gliedert sich in mehrere, sukzessiv durchgeführte Phasen. Zunächst werden die verschiedenen Methoden zur Evaluierung der Benutzbarkeit von Softwaresystemen betrachtet. Anhand dieser Betrachtungen wird dann eine für die im Rahmen dieser Arbeit zu erhebenden Informationen geeignete Methode ausgewählt. Daraufhin werden verschiedene Produkte für die Geschäftsprozessmodellierung sowie die visuelle Erstellung von HTML-Formularen betrachtet. Diese Produkte werden hinsichtlich ihrer Funktionalität und Eignung für die Domäne der einfachen Prozesse in Webportalen kategorisiert. Aus den verschiedenen Kategorien und Ansätzen werden dann Repräsentanten selektiert, deren Eignung und Gebrauchstauglichkeit anhand von Usability-Tests verifiziert wird.

Im Rahmen der Vorbereitungen zu den Usability-Tests wurde für jeden der getesteten Ansätze eine angemessene Aufgabenstellung ausgearbeitet. Da kein direkter Vergleich der Systeme oder deren jeweiliger Funktionsumfang, sondern eine qualitative Evaluierung der Repräsentanten der verschiedenen Ansätze durchgeführt werden sollte, war an dieser Stelle eine einheitliche Aufgabenstellung für alle getesteten Systeme nicht angemessen. Es wurde für jeden Ansatz eine separate Aufgabenstellung erarbeitet, anhand derer die wesentlichen Konzepte und Aspekte des jeweiligen Produkts beleuchtet werden sollten. Da die gewählten Produktansätze sich stark in ihrem Aufbau, Konzept und Funktionen unterscheiden, hätte eine identische Aufgabenstellung bei den unterschiedlichen Ansätzen entweder nicht die gravierenden Aspekte und Funktionen des getesteten Systems herausstellen können, oder hätte an anderer Stelle die Probanden vor unlösbare Aufgaben gestellt. Das würde zu einer unangemessen hohen Anzahl an Frustrationserlebnissen der Probanden führen, was bei der Durchführung von Usability-Tests zu vermeiden ist.

Nach Festlegung der Aufgabenstellungen wurde der Testplan ausgearbeitet. Darauf folgend wurden die räumlichen und technischen Vorbereitungen zur Durchführung der Usability-Tests

getroffen. Es wurde ein Usability-Labor eingerichtet und entsprechende Hard- und Software-Installationen vorgenommen. Während der Durchführung der Tests wurden an dieser Infrastruktur keine Veränderungen vorgenommen. Die Erhebung der Testdaten erfolgte durch das Protokollieren des Testadminstrators im Testplan, sowie über eine Screen-Capture Software, welche zusätzlich zu den Bildschirmaktivitäten die sprachlichen Äußerungen des Probanden aufzeichnete. Die Erhebung der Testdaten erfolgte über die Auswertung des Testplans, sowie der durch die Screen-Capture Software aufgezeichneten Daten. Anhand dieser Auswertungen wurde dann die Analyse der Produktansätze bzgl. ihrer Stärken und Schwächen vorgenommen.

3.1 Beschreibung der Problemdomäne

Die im Kontext dieser Arbeit betrachtete Domäne der einfachen Prozesse in Webportalen umfasst aus Sicht der Domänenexperten das leichte und intuitive Erstellen von HTML-Formularen mit der Möglichkeit der Dynamisierung des Formulars, sowie das Verknüpfen dieser Formulare mit bestimmten Funktionen und Abläufen, welche als leichtgewichtige Geschäftsprozesse zu betrachten sind. Bei diesen einfachen Geschäftsprozessen des Online-Formularwesens stehen Anforderungen wie das Versenden von personalisierten E-Mail-Bestätigungen oder Benachrichtigungen im Vordergrund. Auch die Möglichkeit zu einem bestimmten Zeitpunkt eine Erinnerungs-E-Mail (Reminder) an einen Benutzer eines Online-Formulars zu verschicken oder das Einrichten eines Dispatcher-Mechanismus für eingehende Formulardaten sind Anforderungen im Kontext der genannten Domäne. Ebenso sind die Möglichkeit der Status- und Prioritätenvergabe für eingegangene Formulardaten und das Festlegen von Stellvertreter- und Gruppenregelungen sowie das Definieren von Eskalationsszenarien wichtige Punkte. Die folgenden drei Epics und deren Aufsplittung in mehrere kleine User-Stories innerhalb der jeweiligen Tabelle zeigen die genannten Anforderungen im Anwendungskontext.

User Story 1: Anmeldung zu einer Lehrveranstaltung

Themenschwerpunkte: Datenbank-Anbindung, Reminder-Funktion

Back-End	Front-End
Als Domänenexperte kann ich ein Formular zur Anmeldung für eine Lehrveranstaltung erstellen oder bearbeiten.	Als Student kann ich mich über ein Online-Formular zu einer Lehrveranstaltung anmelden. Nach der Anmeldung bekomme ich eine Anmeldebestätigung.
Als Domänenexperte kann ich • ein Online-Formular erstellen, welches die Anmeldung zu einer Lehrveranstaltung ermöglicht • Informationen zu der Lehrveranstaltung angeben • die Teilnehmerzahl einer Lehrveranstaltung begrenzen • Tutorien zu einer Lehrveranstaltung hinzufügen • festlegen, ob die Teilnahme an einem Tutorium obligatorisch ist • festlegen, ob die Teilnehmer vor Beginn der Lehrveranstaltung erinnert werden sollen • mir alle Teilnehmer einer Lehrveranstaltung anzeigen lassen • eine E-Mail an alle Teilnehmer versenden • ein vorhandenes Formular bearbeiten	Als Student kann ich • meine persönlichen Daten bei der Anmeldung zu einer Lehrveranstaltung eingeben • mir ggf. ein Tutorium zu einer Lehrveranstaltung auswählen • über Änderungen der Lehrveranstaltung informiert werden • mich bei einer ausgebuchten Veranstaltung über frei gewordene Plätze benachrichtigen lassen

Tabelle 3.1: User Story 1: ‘Anmeldung zu einer Lehrveranstaltung’

Abbildung 3.1 zeigt die grafische Modellierung der User-Story ‘Anmeldung zu einer Lehrveranstaltung’ in der Business Process Modeling Notation (BPMN).

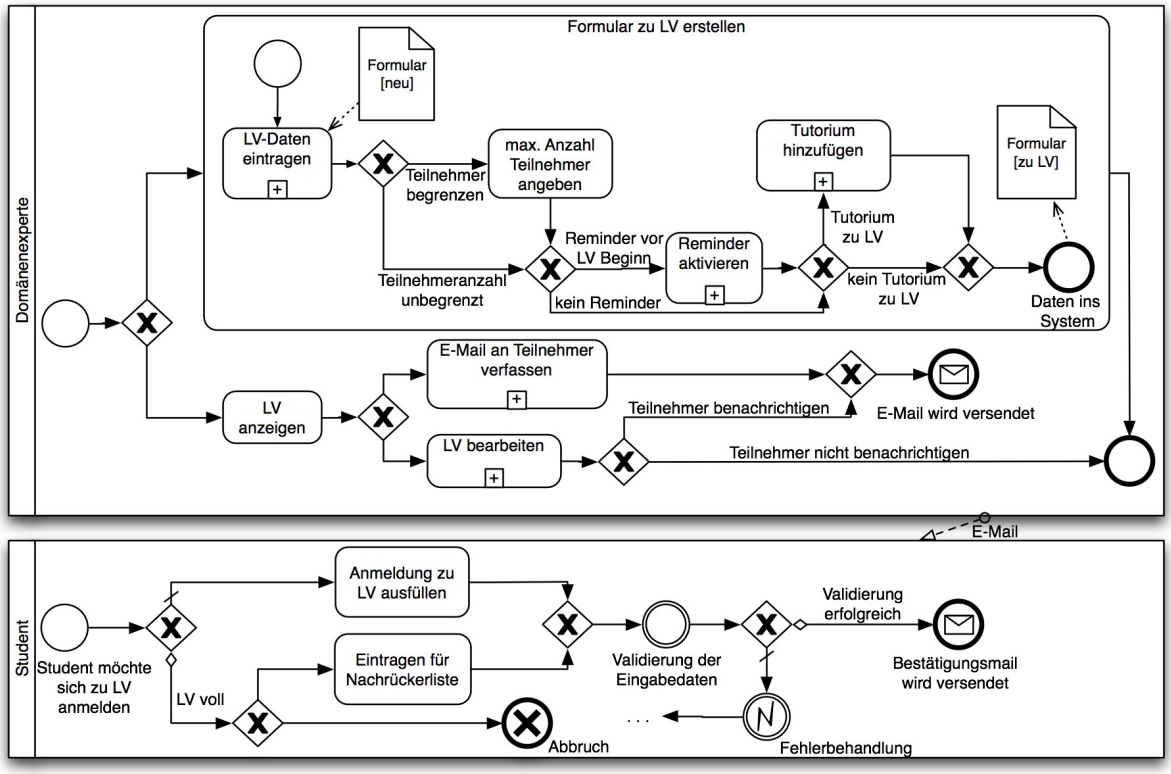


Abbildung 3.1: User Story ‘Anmeldung zu einer Lehrveranstaltung in BPMN’

User Story 2: externe Anfrage über ein Online-Formular

Themenschwerpunkte: Dispatcher, Status, Stellvertreter- und Gruppenregelung

Back-End	Front-End
Als Domänenexperte kann ich ein Online-Formular für eine (externe) Kontaktanfrage anlegen oder bearbeiten. Dabei kann ich Stellvertreter- oder Gruppenregelungen vornehmen.	Als Portalbenutzer kann ich eine Anfrage über ein Onlineformular absenden. Ich werde über den Eingang der Anfrage benachrichtigt.
Als Domänenexperte kann ich • ein Online-Formular für eine externe Anfrage anlegen • ein vorhandenes Anfrageformular bearbeiten • Dipsatching-Regeln für bestimmte Kriterien festlegen • Stellvertreter- oder Gruppenregelungen für die Empfänger festlegen • alle Anfragen an mich bearbeiten	Als Portalbenutzer • kann ich über ein Kontaktformular eine Anfrage stellen • werde ich über den Eingang meiner Anfrage benachrichtigt

Tabelle 3.2: User Story 2: ‘externe Anfrage’

Abbildung 3.2 zeigt die grafische Modellierung der User-Story ‘externe Anfrage über ein Online-Formular’ in BPMN.

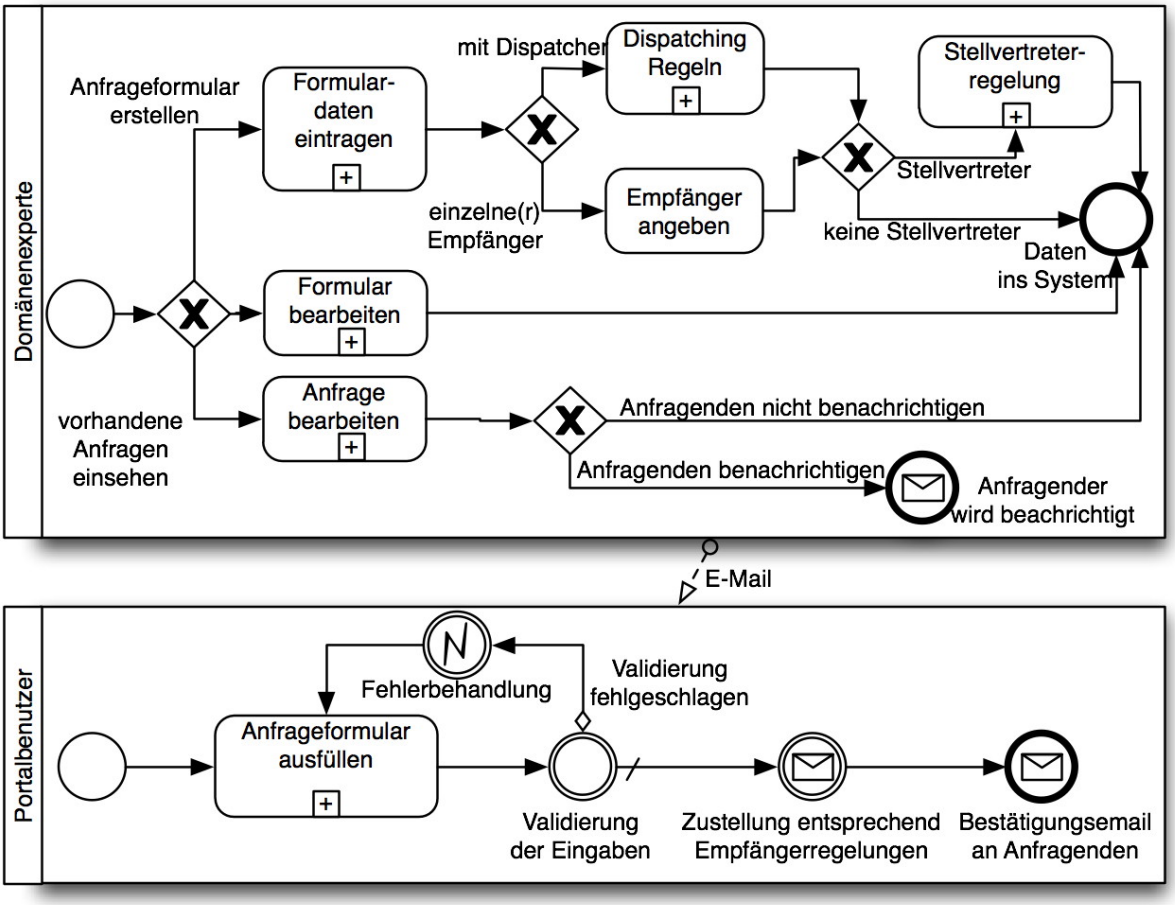


Abbildung 3.2: User Story ‘externe Anfrage über ein Online-Formular’ in BPMN

User Story 3: interne Anfrage über ein Online-Formular

Themenschwerpunkte: Eskalation, Prioritäten

Back-End	Front-End
Als Domänenexperte kann ich für interne Portalbenutzer (mit Benutzerkennung) ein Formular für interne Anfragen bereitstellen und für eingehende Anfragen ein Eskalationsszenario definieren.	Als interner Portalbenutzer kann ich eine interne Anfrage über ein Online-Formular stellen. Ich werde über den Eingang der Anfrage benachrichtigt und kann auf einer Webseite alle meine Anfragen und deren Status einsehen und kommentieren.
Als Domänenexperte kann ich • ein Online-Formular für eine interne Anfrage mit unterschiedlichen Feldern und Optionen anlegen • in dem Formular eine Vergabe von Prioritäten ermöglichen • ein Eskalationsszenario definieren • ein vorhandenes Anfrageformular bearbeiten • alle Anfragen eines Benutzers einsehen • mich über neue Kommentare eines Benutzers zu seiner Anfrage informieren lassen	Als interner Portalbenutzer kann ich • eine Anfrage an einen internen Bereich stellen • eine interne Anfrage mit einer Priorität versehen • eine Benachrichtigung über den Eingang meiner Anfrage erhalten • auf einer Webseite meine Anfragen einsehen und kommentieren

Tabelle 3.3: User Story 3: ‘interne Anfrage’

Abbildung 3.3 zeigt die grafische Modellierung der User-Story ‘interne Anfrage über ein Online-Formular’ in BPMN.

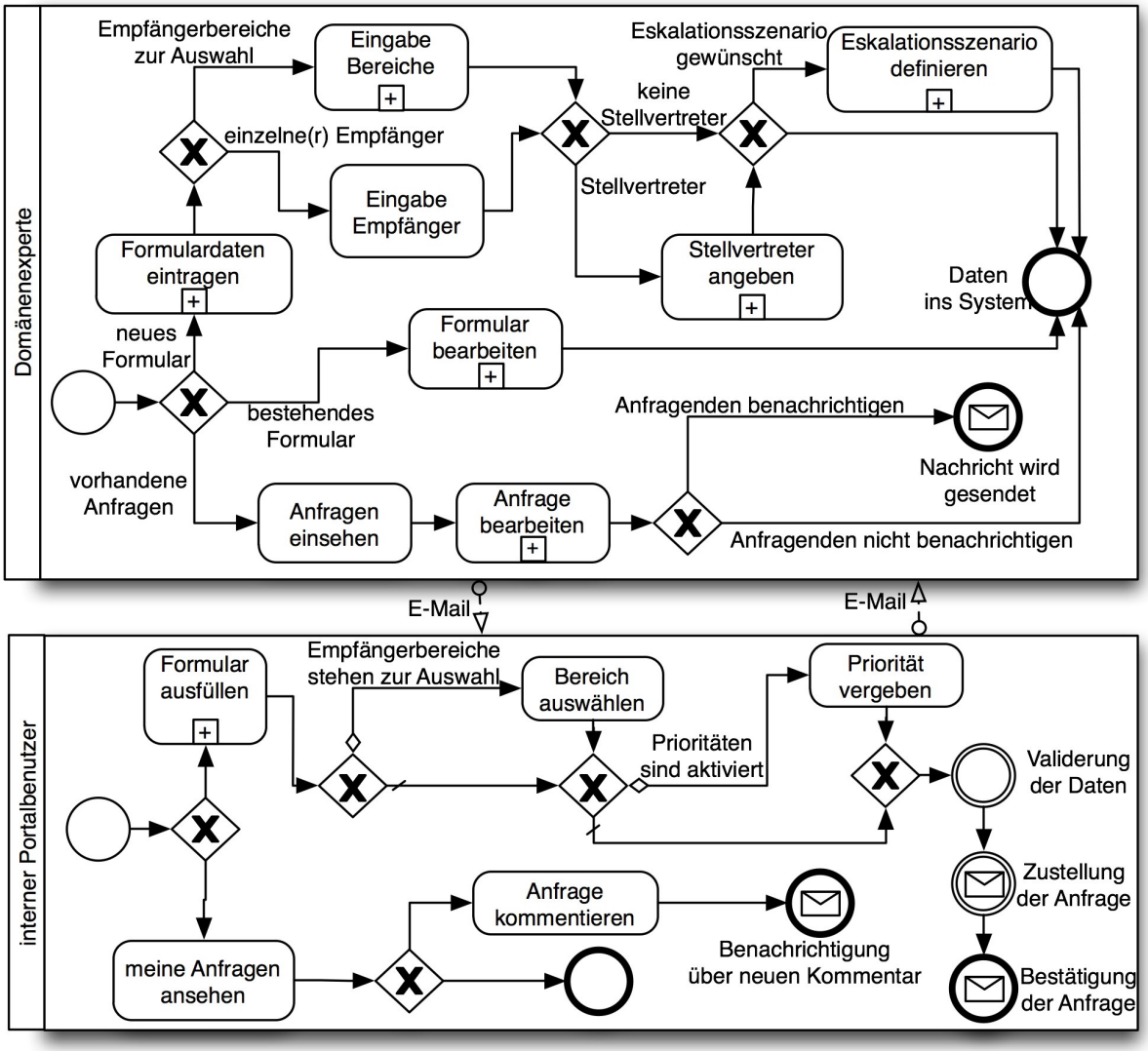


Abbildung 3.3: User Story ‘interne Anfrage über ein Online-Formular’ in BPMN

3.2 Bestehende Systeme

Die Betrachtung der bestehenden Systeme im Bereich des visuellen End User Development für das Workflow- und Formularwesen hat ergeben, dass diese in drei Kategorien einzuteilen sind:

1. In die erste Kategorie fallen die so genannten visuellen Formulargeneratoren. Hierbei handelt es sich um Systeme für das visuelle Zusammenstellen von HTML-Formularen. Produkte dieser Art sind für diesen speziellen Zweck gut geeignet, bieten aber keine oder nur sehr geringe Möglichkeiten zur Abbildung von Geschäftsprozessen mit den Formularen.

2. In der zweiten Kategorie finden sich die Produkte für die Geschäftsprozessmodellierung und -optimierung, welche umfangreiche Softwaresuiten zur Abbildung komplexer Vorgänge sind. Produkte dieser Art unterstützen nicht die Erstellung von Web-Formularen und sind nicht für die Problemdomäne der unternehmensinternen oder -übergreifenden Prozesse in Webanwendungen gemacht.

3. Die dritte und im Rahmen dieser Arbeit interessante Produktkategorie umfasst Systeme, die sowohl die Möglichkeit der Erstellung von Formularen, als auch deren Verknüpfung mit Workflow-Funktionen bieten.

Im Anhang 11 befindet sich eine kategorisierte Übersicht der betrachteten Produkte aus den jeweiligen Kategorien und deren jeweiligen Eigenschaften und Funktionen.

3.3 Produkte für die Usability-Tests

Im Rahmen der Usability-Tests wurden drei unterschiedliche Produktansätze untersucht. Die in dieser Arbeit betrachtete Problemdomäne umfasst sowohl das leichte und intuitive Erstellen eines HTML-Formulars durch den Domänenexperten, als auch das Verknüpfen dieses Formulars mit bestimmten Funktionen. Daher wurden zwei Produkte jeweils als Repräsentant der ersten und dritten Kategorie ausgewählt. Um einen alternativen, nicht visuellen Ansatz zu evaluieren, wurde eine domänenspezifische Sprache als Repräsentant für den Ansatz der verbalen Notation von Formularen und deren Verbindung mit Workflows gewählt.

Als Referenzprodukt der ersten Kategorie wurde die Rich Internet Application *jotform* [1] ausgesucht. Das Produkt ist ein WYSIWYG Formulargenerator. Durch die Kombination des Drag-and-Drop Bedienungskonzepts mit einer sehr klar strukturierten Oberfläche, welche jederzeit eine Vorschau des Formulars beinhaltet, verspricht es eine leichte und intuitive Bedienung für den Benutzer. Die erstellten Formulare werden vom Anbieter gehostet und es

[1] siehe http://jotform.com/

ist möglich, als Domänenexperte eine E-Mail-Benachrichtigung zu aktivieren, welche einen über eingegangene Formulardaten informiert oder sich die eingegangenen Formulardaten in einer Übersicht anzeigen zu lassen. Konzeptionell ist das Produkt so aufgebaut, dass das Formular die einzelnen Formularelemente (Texteingabefelder etc.) enthält und dass zwischen den Eigenschaften der einzelnen Formularelemente und den Eigenschaften des gesamten Formulars unterschieden wird.

Als Repräsentant der dritten Kategorie wurde die Online-Anwendung *formdesk* [2] gewählt, welche ein Formular-Management System mit Workflow-Unterstützung ist. Wie es auch bei den meisten Produkten zur Geschäftsprozessmodellierung erforderlich ist, empfiehlt die Anwendung eine Modellierung des Workflows in der BPMN, bevor dieser mit dem System umgesetzt wird. Das System ermöglicht es dem Domänenexperten E-Mail-Benachrichtigungen zu aktivieren, welche über eingegangene Formulardaten informiert. Bei diesen E-Mail-Benachrichtigungen können unterschiedliche, in Abhängigkeit von den ausgefüllten Formulardaten definierbare, Kriterien für den Versand und die Empfänger definiert werden. Es ist außerdem möglich, personalisierte E-Mail-Bestätigungen an den Benutzer des Formulars zu senden. Auch kann das Formular insofern dynamisiert werden, dass es möglich ist, Formularelemente in Abhängigkeit der Werte anderer Formularelemente anzuzeigen, auszublenden, zu aktivieren oder zu deaktivieren. Besonders geeignet erschien dieses Produkt unter anderem aufgrund der Tatsache, dass es weitreichende Funktionen für das betrachtete Problemfeld der Prozesse in Webportalen unterstützt und dabei hervorhebt, dass keinerlei Vorkenntnisse der Benutzer zur Umsetzung ihrer Anforderungen und der Formular-Workflows notwendig sind.

Um den Ansatz der verbalen Notation bzgl. der Benutzbarkeit für Benutzer zu evaluieren, wurde die domänenspezifische Sprache *WebDSL* [3] gewählt. Die Sprache unterstützt Methoden für die Domänenmodellierung, die Präsentationsschicht, den Seitenablauf, Zugriffskontrolle und Workflows und verwendet dabei Begriffen, die an die Bezeichnung von HTML-Elementen und Internetseiten angelehnt sind.

[2] siehe http://jotform.com/
[3] siehe http://webdsl.org/webdslorg/home

4 Durchführung der Usability-Tests

Nachdem eine Abgrenzung der Anwendungsdomäne und eine Auswahl repräsentativer Produkte für diese Domäne vorgenommen wurde, geht es nun darum, diese gewählten Produkte bzgl. ihrer Usability zu evaluieren.

An dieser Stelle soll zunächst die Bedeutung des Begriffs Usability noch einmal durch ein Zitat von [Ram] veranschaulicht werden:

"Um den essentiellen Charakter zu erkennen, sollte man sich dem Begriff über den englischen Wortstamm nähern:

to use benutzen, gebrauchen, verwenden

the ability die Möglichkeit

[...] Da das reine Erkennen der Notwendigkeit zur kompromisslosen Ausrichtung an den Bedürfnissen der Benutzer nicht ausreicht, sondern auch gehandelt werden muss, steckt im Wortstamm des Begriffs eine Handlungsaufforderung:

Use the ability = "Nutze die Möglichkeiten"

Die wohl treffendste Übersetzung für den Begriff Usability ist "Gebrauchstauglichkeit" die auch in Normen und Standards verwendet wird."

Wie in [geb] beschrieben, charakterisiert die Gebrauchstauglichkeit eines Softwareprodukts dessen Eignung, in einem bestimmten Anwendungskontext genutzt werden zu können, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen.

Weiterhin wird in [geb] erläutert, dass zur Gewährleistung einer guten Usability der wesentliche Faktor darin besteht, dass die Software für den Benutzer leicht verständlich und schnell benutzbar sein muss. Diese beiden genannten Aspekte fallen in den Bereich der Software-Ergonomie, deren Aspekte in [swe] genannt und dort mit dem folgenden Zitat zusammengefasst sind:

"Software-Ergonomie (Abk. SE, v. griech.: ergon = "Werk" (Mühe, Arbeit) + nomos = Lehre, Gesetz, Regel / engl.: Usability Engineering) ist die Arbeit hin zu leicht verständlicher und schnell benutzbarer Software unter den gebotenen technischen Möglichkeiten und unter der Einhaltung definierter bzw. empirisch entstandener Standards und Styleguides".

Im Fokus der Software-Ergonomie steht also der mit der Software interagierende Mensch und die bei dieser Interaktion zu berücksichtigenden psychologischen Aspekte, also der Anpassung von Benutzerschnittstellen an die kognitiven und physischen Fähigkeiten bzw. Eigenschaften des Menschen.

Um Erkenntnisse bezüglich der Benutzbarkeit bestehender Softwaresysteme für die Domäne der Prozesse in Web Portalen zu gewinnen, wurden im Rahmen dieser Arbeit eine qualitative Analyse einer Auswahl dieser bestehenden Softwareprodukte durchgeführt. Da das Usability-Testing laut [Sim] eine im Rahmen der empirischen Softwareevaluation einzuordnende, gängige Technik zur Bewertung der ergonomischen Softwarequalität ist, wurde diese Evaluationstechnik gewählt und mit den Domänenexperten durchgeführt.

4.1 Konzepte von Usability-Tests

Der Fokus von Usability-Tests liegt auf der Evaluation eines bestehenden User-Interfaces bzgl. der mentalen Vorstellungen der Benutzer. Dafür gibt es gibt verschiedene, in [met] aufgeführte Methoden, deren zentrale Punkte im Folgenden kurz erläutert werden.

4.1.1 Interaktionsaufzeichnung

Bei der Interaktionsaufzeichnung steht die Aufzeichnung des Umgangs des Benutzers mit den Eingabegeräten wie Tastatur und Maus im Zusammenhang mit der zu testenden Oberfläche im Fokus. Der Benutzer selbst wird nicht aufgezeichnet und ist auch nicht Bestandteil der Testauswertung. Das Ergebnis der Testdurchführung ist somit eine Aufzeichnung der Benutzeroberfläche und der darauf stattfindenden Aktionen während der Aufgabendurchführung [met].

4.1.2 Videobeobachtung

Die Methode der Videobeobachtung ist eine Erweiterung der Interaktionsaufzeichnung, bei der zusätzlich der Proband während der Aufgabendurchführung aufgezeichnet wird. Hier fließt also das Verhalten des Probanden mit in die Auswertung der Testdurchführung ein. Das Aufzeichnen des Testteilnehmers erfordert eine relativ umfangreiche Video-Ausrüstung sowie eine aufwendige Installation des Equipments, daher werden Usabilty-Tests in Form von Videobeobachtungen bevorzugt in entsprechend ausgestatteten Usability-Laboren durchgeführt [met]. Zudem gestaltet sich die Auswertung der Testdurchführungen relativ komplex, da die Aktionen der Benutzeroberfläche korrelativ mit dem aufgezeichneten Verhalten des Benutzers ausgewertet werden müssen.

4.1.3 Blickbewegungsregistrierung (Eye Tracking)

Bei dieser Methode werden die Blickbewegungsrichtungen und -intervalle des Probanden während der Aufgabendurchführung aufgezeichnet. Dadurch ist es möglich, Aussagen über die Wahrnehmung des Benutzers bezüglich bestimmter Bedienelemente auf der Oberfläche treffen zu können [met]. Auch diese Methode erfordert für die Testdurchführung geeignete Aufzeichnungsgeräte sowie deren sachgemäße Installation und ist daher für die Durchführung in einem Usability-Labor geeignet.

4.1.4 Lautes Denken (Thinking Aloud)

Diese Methode ist die am meisten verbreitete und zudem eine sehr effektive Methode des Usability-Testens. Zentraler Aspekt zur Erhebung der Testdaten sind hier lautsprachliche Äußerungen des Probanden während der Interaktion mit der Anwendung [met]. Anhand dieser Äußerungen des Probanden bzgl. seiner Gefühle und Probleme während der Bearbeitung einer Aufgabe kann festgestellt werden, wie der Proband mit der Anwendung interagiert und wo Probleme welcher Art auftreten. Bei dieser Methode des Usability-Testens sind verschiedene Stufen der Aufzeichnung möglich. Je nach Einsatzgebiet kann eine allein durch Notizen des Testadministrators begleitete Testdurchführung bis hin zur kompletten Video- und Audioaufzeichnung der Oberfläche und des Probanden angemessen sein. In Abhängigkeit der Komplexität der erhobenen Testdaten wächst der für die Auswertung benötigte Aufwand.

4.1.5 Usability-Inspektionsmethoden

Usability-Inspektion wird nicht durch Benutzer der Zielgruppe der Software durchgeführt, sondern durch Experten, welche Endanwender, Domänenexperten, Softwareentwickler oder Usability-Ingenieure sein können [Eicc]. Um die gravierenden bzw. offensichtlichen Usability Probleme einer Anwendung im Vorhinein zu erkennen und zu beheben, sollte vor den eigentlichen Usability-Tests eine Usability-Inspektion durch Experten stattfinden [kog]. Dabei baut dieser Ansatz auf die Fähigkeit der Experten, die Probleme der Endanwender vorherzusagen [Eicc].

Eine Form der Usability-Inspektion ist die heuristische Evaluation, bei der die Experten das Produktinterface unter Betrachtung bestimmter Usability-Prinzipien (Heuristiken) untersuchen [Eicb].

Der kognitive Durchlauf ist eine szenariobasierte Usability-Inspektionsmethode. Die Experten erkunden unabhängig voneinander die aus Benutzersicht relevanten Funktionen des

Produkts. Währenddessen werden von anderen Experten auftretende Schwierigkeiten notiert [met]. Es werden also keine empirischen Benutzerdaten sondern die Urteile eines oder mehrerer Experten erfasst [Eica].

4.1.6 Moderierter Gruppentest

Diese Methode basiert auf der Audioaufzeichnung einer Gruppe von Benutzern während der Bearbeitung von Fragen und Aufgabenstellungen zu dem untersuchten Produkt [met]. Am Ende werden die einzelnen Äußerungen verglichen und Übereinstimmungen notiert.

4.2 Auswahl einer Usablility-Testmethode

Aus denen im vorhergehenden Abschnitten vorgestellten Konzepten für Usability-Tests muss nun eine für die im Rahmen dieser Arbeit durchzuführenden Tests eine adäquate Testmethode ausgewählt werden.

Die Methode 'Lautes Denken' wird in der Softwareentwicklung sehr effektiv zur Erhebung qualitativer Informationen bzgl. der Angemessenheit und Benutzbarkeit von Software eingesetzt [Fro]. Die Testpersonen führen typische Aufgaben mit dem Produkt aus, und schon mit einer geringen Anzahl an Testpersonen werden sehr gute Ergebnisse erzielt, welche aufzeigen an welchen Stellen es zu Problemen und Schwierigkeiten in der Benutzung kommt [Str]. Die Testaufzeichnung kann sich je nach situativen Gegebenheiten variabel gestalten und die Auswertung der erhobenen Daten kann relativ schnell und unkompliziert erfolgen [Str].

Die Qualität dieses Testverfahrens ist stark abhängig von den differenzierten Äußerungen der Probanden über ihr Denken und Handeln. Bei einem Test werden nicht nur die Usability-Probleme aufgedeckt, sondern durch die vom Benutzer spontan artikulierten Gefühle und Gedanken bei seinen Aktionen ist es zusätzlich möglich zu ergründen, warum diese Probleme auftraten [Str].

Zu beachten ist, dass die Versuchspersonen eine für die Zielgruppe der Software repräsentative Auswahl darstellen und ihre Kenntnisse einen leistungsheterogenen Querschnitt bilden sollten [Fro].

Bei der Auswertung darf nicht außer Acht gelassen werden, dass das Artikulieren des Probanden während der Durchführung von Aktionen zu einer Verzögerung ihrer Arbeitsgeschwindigkeit führen kann. Auch kann es sein, dass durch das Mitsprechen der Probanden ihr Verhalten in Problemsituationen positiv beeinflusst wird [Str].

4.3 Anzahl der Probanden

Um durch die Usability-Tests ausreichende und aussagekräftige Testdaten unter einem vertretbaren Aufwand zu erhalten, ist es notwendig, sich über die Anzahl der eingesetzten Probanden Gedanken zu machen.

Die Anzahl der Testpersonen wurde basierend auf den Ergebnissen der Studie von Tom Landauer und Jakob Nielsen [NL93] gewählt. Diese besagt, dass eine Testdurchführung mit 5-6 Benutzern völlig ausreichend ist, um die meisten Usability-Schwachstellen aufzudecken. Im Rahmen dieser Studie fanden Landauer und Nielsen heraus, dass die Zahl der Usability-Schwachstellen, die in einem Usability-Test mit n Benutzern entdeckt werden, sich annähernd durch folgende Formel berechnen lässt:

$$N(1-(1-L)^n) \tag{4.1}$$

n steht für die Anzahl der Benutzer, N für die Anzahl aller Usability-Probleme und L ist der durch den jeweiligen Benutzer entdeckte Anteil der Usability-Probleme. Der Wert von L liegt üblicherweise bei 31%, damit ergibt sich für die Formel die in Abbildung 4.1 gezeigte Problemfindungskurve.

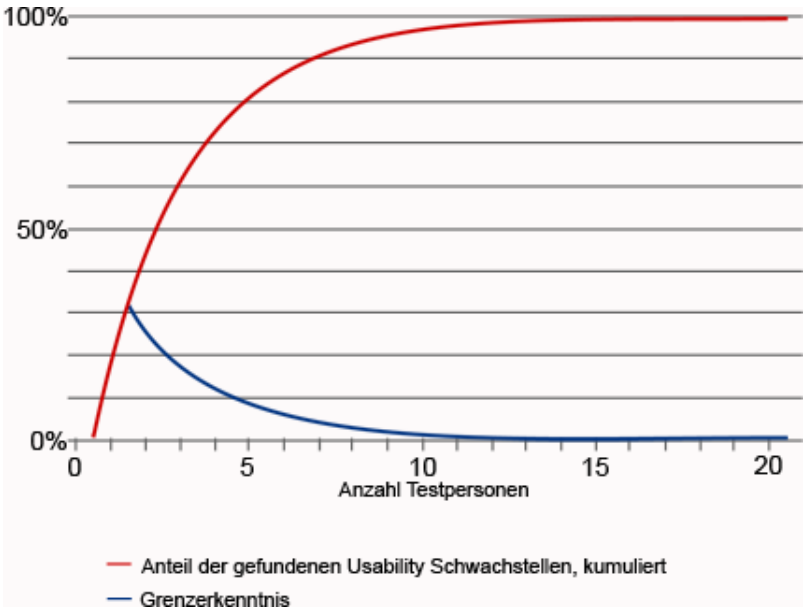


Abbildung 4.1: Problemfindungskurve bei Usability-Tests, Quelle:[Ble]

Aus Abbildung 4.1 lassen sich nach [Ble] folgende Aussagen herleiten :

- Trivialer Weise wird durch einen Verzicht auf einen Usability-Test keines der Usability-Probleme erkannt.
- Eine Testdurchführung mit einem Probanden deckt 31% der Usability-Probleme auf.
- Durch den zweiten Benutzer werden weitere 21% der Usability-Schwachstellen des Systems entdeckt.
- Bei der dritten Testperson kommen noch einmal 15% zu den entdeckten Usability-Schachstellen hinzu.
- Mit wachsender Benutzeranzahl wird der Erkenntniszuwachs pro durchgeführtem Test immer kleiner.
- 85-89% aller Usability-Schwachstellen sind ab dem 5.- 6. Benutzer erkannt.
- Ab dem 6. Benutzer steigt das Kosten-Nutzen Verhältnis stark an und die Testdurchführung wird damit unökonomischer. Denn um die weiteren 10-15% aller Usability-Probleme aufzudecken wären ca. 10 weitere Testdurchführungen notwendig.

Entsprechend dieser Betrachtungen wurden für die Usability-Tests im Kontext dieser Arbeit 5 Testpersonen eingesetzt.

4.4 Das Usability-Labor

Nachdem in den vergangenen Abschnitten die theoretischen Überlegungen und Vorbereitungen zu den Testdurchführungen getroffen wurden, mussten nun die räumlichen und technischen Voraussetzungen geschaffen werden.

Die Usability-Tests im Rahmen dieser Arbeit wurden in einem Meetingraum der Infopark AG durchgeführt. In diesem Raum befand sich ein Tisch, auf dem die Test-Infrastruktur aufgebaut wurde. Die Test-Infrastruktur wurde so aufgebaut, dass der Proband und der Testadministrator an einer langen Seite des Tisches nebeneinander saßen, wobei der Testadministrator links vom Probanden saß. Abbildung 4.2 zeigt das Usability-Labor mit der aufgebauten Infrastruktur.

Abbildung 4.2: Das Usability-Labor für die Durchführung der Usability-Tests

Die hardwaretechnische Test-Infrastruktur bestand aus einem Apple MacBook (2,1 GHz Intel Core 2 Duo, 2 GB Arbeitsspeicher, Betriebssystem Mac OS X 10.5.6), an das ein externer 17"Monitor mit einer Auflösung von 1280 x 1024 Pixeln sowie eine externe Tastatur und Maus angeschlossen waren. Für die Audioaufzeichnung der Tests war ein Mikrofon angeschlossen, welches direkt unter dem externen Monitor vor dem Probanden platziert war. Die Internetanbindung erfolgte über Wireless LAN. Der Arbeitsplatz des Probanden bestand somit aus dem externen Monitor, welcher als Hauptbildschirm eingestellt war, der externen Tastatur und der externen Maus.

Die Testdurchführung erfolgte in einer ruhigen und störungsfreien Umgebung, damit es zu keinerlei Beeinflussung oder Störung der Testperson während der Testdurchführung kommen konnte. Während der Testdurchführung wurde im Hintergrund ein Screen-Capture Programm mit Audioaufnahme gestartet, welches sämtliche auf dem externen Monitor stattfindenden Aktivitäten sowie die Äußerungen des Benutzers aufzeichnete. Im Fokus der Evaluierung des einzusetzenden Screen-Capture Produkts stand die Nicht-Beeinflussung der Testdurchführung durch die Screen-Capture Software, sodass ein Produkt gewählt werden musste, welches diskret und ressourcenschonend im Hintergrund arbeitet, so dass es zu keiner extremen Systemauslastung kommt, welche eine Reaktionsverzögerung des Systems nach sich ziehen könnte.

4.5 Der Testplan

Der Testplan ist ein Dokument, das bei jedem durchzuführenden Usability-Test als Leitfaden für den Testadministrator dient. Er gibt den Ablauf der Testdurchführung vor und dient zur Erhebung der Daten zur Testperson und zur Dokumentation während der Testdurchführung. Der Testplan, der für die im Rahmen dieser Diplomarbeit durchzuführenden Usability-Tests entwickelt wurde, enthält als Erstes den Abschnitt zur Erhebung des Userprofils des Probanden. In diesem werden die folgenden Informationen erfasst:

- Name
- Alter
- Geschlecht
- Ausbildung
- beruflicher Tätigkeitsbereich
- hauptsächliche Nutzung des Computers
- Erfahrung im Bereich Web-Programmierung
- Erfahrung mit Produkten zur Prozessmodellierung

Außerdem enthält der Testplan eine vom Probanden zu unterzeichnende Einverständniserklärung, eine Übersicht der technischen Test-Infrastruktur, eine kurze Information zu den zu testenden Produkten, einen kurzen Einführungstext für den Probanden und die Aufgaben des Beispieldurchlaufes, welcher zur Eingewöhnung des Probanden an das Testverfahren vor Beginn der eigentlichen Tests durchgeführt wird.

Für jedes zu testende Produkt enthält der Testplan die gesamte Aufgabenstellung, die einzelnen vom Probanden durchzuführenden Tasks und die Fragen des anschließenden Interviews. Wie Abbildung 4.3 zeigt, gibt es pro Task eine Beschreibung des Tasks, eine Beschreibung des O.K.-Kriteriums für diesen Task, sowie Felder, in denen während der Durchführung Bemerkungen des Testadministrators und die vom Probanden für diesen Task benötigte Zeit und die Erfüllung des Tasks festgehalten werden.

Task 3		**Anrede**
Beschreibung	Fügen Sie dem Formular für die Anrede zwei beschriftete Radio-Buttons hinzu. Dabei soll der Wert „Herr“ die Vorauswahl sein. Die Angabe der Anrede ist ein Pflichtfeld und als solches gekennzeichnet.	
O.K. Kriterium	Das Formular enthält zwei Radio-Buttons zur Auswahl der Anrede, wobei der Button mit dem Wert „Herr“ vorselektiert ist.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Abbildung 4.3: Repräsentation eines Tasks im Testplan

4.6 Durchführungsbeschreibung

Insgesamt wurde bei der Durchführung der Usability-Tests versucht, eine entspannte und für die Probanden angenehme Atmosphäre zu schaffen. Es war deutlich zu spüren, dass zunächst bei den Probanden große Anspannung und teilweise Nervosität herrschte, da sie sich einer Prüfungssituation ausgesetzt fühlten. Daher wurde vom Testadministrator zunächst durch Ablenkungen, wie z.B. kurze Gespräche in einem anderen Kontext oder beispielsweise die Frage nach einem Getränkewunsch versucht, sich aus der Position des 'Prüfenden' zu rücken und zur emotionalen Entspannung der Probanden beizutragen. Dann wurde den Probanden der Zweck und die zentralen Aspekte von Usability-Tests erläutert. Insbesondere wurde die in diesen Tests eingesetzte Methode des 'Thinking Aloud Usability-Testing' erklärt und betont, dass diese Methode von dem verbalen Feedback des Probanden bzgl. seiner Gedanken und Gefühle während der Aufgabendurchführung lebt.

Um dem Probanden weitere Anspannungen und das Gefühl einer Prüfungssituation zu nehmen, wurde deutlich gemacht, dass nicht er als Proband getestet wird, sondern die jeweiligen Softwareprodukte, und dass dafür seine Mitwirkung von großer Bedeutung ist. Es wurde hervorgehoben, dass es in keiner Weise darum geht, seine Art und Weise der Umsetzung oder seine Durchführungsgeschwindigkeit bei einer Aufgabe zu bewerten. Vielmehr ist es das Ziel, die jeweiligen Stärken und Schwächen der einzelnen Systeme sowie deren Eignung für den Bereich der Modellierung von Prozessen in Webportalen zu erkennen. Danach wurde dem Probanden die technische Infrastruktur sowie die Auswahl der zu testenden Produkte erläutert und er bekam die Möglichkeit, Fragen an den Testadministrator zu stellen.

Waren alle Fragen des Probanden geklärt, so wurde mit der Abarbeitung des Testplans begonnen, was zunächst die Erhebung der Informationen für das Benutzerprofil des Probanden sowie dessen Unterzeichnung der Einverständniserklärung beinhaltete. Danach sollte sich der Proband anhand eines Probedurchlaufs, welcher das Zusammenbauen einer Legofigur in der Anwendung 'Lego Digital Designer' forderte, an die Art der Testdurchführung sowie das Äußern seiner Gefühle und Gedanken während der Bearbeitung von Aufgaben gewöhnen.

Anschließend wurde mit der Durchführung der eigentlichen Tests begonnen. Für jedes zu testende Produkt bekam der Proband zuerst die entsprechende Aufgabenstellung vorgelegt, welche sich dieser durchlesen sollte um anschließend aufkommende Verständnisfragen oder Unklarheiten mit dem Testadministrator zu klären. War die Aufgabenstellung klar, so wurde dem Probanden die Liste der einzelnen Tasks vorgelegt. Der Ablauf eines einzelnen Tasks begann damit, dass der Proband den Task durchlas und dem Testadministrator mitteilte, wenn er mit der Umsetzung begann. Während der Durchführung des Tasks verfolgte der Testadministrator das Vorgehen und machte sich entsprechende Notizen im Testplan. Der Proband teilte dem Testadministrator mit, wenn er der Ansicht war, dass der Task umgesetzt und abgeschlossen sei.

Nach Abschluss des Tests wurde mit dem Probanden ein halbstrukturiertes Interview durchgeführt, in dem er seine Eindrücke während der Testdurchführung reflektieren und sich zu den empfundenen Vor- und Nachteilen des Produkts äußern sollte.

4.7 Inhalte der Usability-Tests

Die Testaufgaben der einzelnen Produkte waren so ausgelegt, dass sie die für die Domäne der Prozessmodellierung in Webportalen relevanten Funktionen nutzen, um so die Eignung der Produkte für diese Domäne zu untersuchen.

4.7.1 *jotform* Formulargenerator

Bei dem WYSIWYG Formulargenerator *jotform* stand das Erstellen eines einfachen Formulars durch den Domänenexperten im Vordergrund. Es sollte getestet werden, wie der Benutzer mit dem Bedienungskonzept und der Oberflächenstrukturierung des Produkts zurecht kommt, indem er dem Formular Elemente hinzufügen sowie deren Eigenschaften (Werte und Optionen) bearbeiten sollte. Durch Aufgaben, die eine Unterscheidung zwischen den Eigenschaften von Formularelementen und den Eigenschaften des gesamten Formulars forderten, sollte der konzeptionelle Aufbau des Produkts verstanden werden.

Für den Formulargenerator *jotform* war die Aufgabenstellung folgend:

- Erstellen Sie ein Kontaktformular mit dem sich ein Kunde zu einer von mehreren Veranstaltungen Ihres Unternehmens anmelden kann. Eine Veranstaltung findet an mehreren unterschiedlichen Terminen statt.
- Das Kontaktformular soll die folgende Funktionalität bieten:
  - zwei Radio-Buttons für die Anrede (Pflichtfeld), wobei der Wert ‘Herr’ vorausgewählt ist
  - ein beschriftetes Eingabefeld für den Namen (Pflichtfeld)
  - ein beschriftetes Eingabefeld für den Vornamen (optionale Angabe)
  - ein beschriftetes Eingabefeld für die E-Mail-Adresse (Pflichtfeld, nur gültige E-Mail-Adressen der Form xxx@yyy.zzz erlaubt)
  - ein mehrzeiliges Eingabetextfeld für freien Nachrichtentext (optionale Angabe)
  - ein Drop-down Auswahlbox mit der der Kunde aus mehreren angebotenen Veranstaltungen eine Veranstaltung auswählen kann, zu der er sich anmelden möchte (Pflichtfeld).

  - wählt der Kunde eine Veranstaltung aus, so sollen in einer weiteren Auswahlbox nur die Termine für die ausgewählte Veranstaltung angezeigt werden. (Pflichtfeld)

- Ein Kunde soll ein unvollständig ausgefülltes Formular nicht abschicken können.

- Der Kunde soll, nachdem er seine Anmeldung vollständig ausgefüllt und abgeschickt hat, eine Bestätigungs-E-Mail mit den Daten der Veranstaltung, zu der er sich angemeldet hat, bekommen.

- Die Daten aus dem Formular sollen per E-Mail an zwei unterschiedliche Empfänger aus ihrem Unternehmen gesendet werden.

Daraus ergaben sich für den Probanden folgende durchzuführende Tasks:

1. Loggen Sie sich auf in dem *jotform*-Formulargenerator ein. Orientieren Sie sich auf der Oberfläche der Anwendung und verschaffen Sie sich einen Überblick über die Funktionen.

2. Erstellen Sie ein neues leeres Formular mit dem Default Style.

3. Fügen Sie dem Formular für die Anrede zwei beschriftete Radio-Buttons hinzu. Dabei soll der Wert 'Herr' die Vorauswahl sein. Die Angabe der Anrede ist ein Pflichtfeld und als solches gekennzeichnet.

4. Fügen Sie dem Formular zwei beschriftete einzeilige Texteingabefelder für Name und Vorname hinzu. Das Feld für den Namen soll ein Pflichtfeld und als solches gekennzeichnet sein. Die Texteingabefelder sollen eine Breite von 30 Zeichen haben. Die maximale Anzahl einzugebender Zeichen ist ebenfalls auf 30 Zeichen beschränkt.

5. Fügen Sie dem Formular ein beschriftetes einzeiliges Texteingabefeld für die E-Mail-Adresse hinzu. Das Feld soll ein Pflichtfeld und als solches gekennzeichnet sein. Das Texteingabefeld soll eine Breite von 30 Zeichen (auch maximale Anzahl Zeichen) haben. Legen Sie fest, dass vor dem Abschicken des Formulars die eingegebene E-Mail-Adresse auf eine gültige Form überprüft wird.

6. Fügen Sie dem Formular ein mehrzeiliges Eingabetextfeld für die optionale Mitteilung hinzu. Das Texteingabefeld soll eine Breite von 30 Zeichen und eine Höhe von 10 Zeichen haben. Das Feld ist kein Pflichtfeld. Es soll mit dem folgenden Text vorbelegt sein: 'Hier können Sie uns eine Nachricht hinterlassen.'

7. Fügen Sie dem Formular die beschriftete Drop-Down-Auswahlbox für die Veranstaltungstitel hinzu. Tragen Sie drei Werte für Veranstaltungstitel ein. Es soll nur ein Veranstaltungstitel auswählbar sein. Die Auswahl eines Veranstaltungstitels ist ein Pflichtfeld.

8. Fügen Sie dem Formular die beschriftete Drop-Down-Auswahlbox für die Veranstaltungstermine hinzu. Ein Veranstaltungstitel hat mehrere Termine. Es sollen nur die zu einem Veranstaltungstitel gehörigen Termine auswählbar sein. Die Auswahl eines Veranstaltungstermins ist Pflicht.

9. Beschriften Sie den Button zum Abschicken des Formulars mit 'Anmelden'.

10. Legen Sie fest, dass ein ausgefülltes Formular an zwei E-Mail-Empfänger (eins@infopark.de und zwei@infopark.de) aus Ihrem Unternehmen geht.

11. Legen Sie fest, dass der Kunde nach Abschicken des Formulars eine Bestätigungs-E-Mail mit seinen im Formular angegebenen Daten erhält.

Das anschließende Interview umfasste als Einstieg die folgenden Fragen:

- Was sagen Sie spontan zu dem Produkt?

- Was sind Ihrer Meinung nach die Stärken und Schwächen des Produkts?

Weitere Fragen/Themen, die ggf. durch die Einstiegsfragen nicht betrachtet wurden:

- Nennen Sie bitte jeweils ein paar Dinge, die Ihnen an dem Programm gut/weniger gut gefallen haben

- Beschreiben Sie bitte Ihre Schwierigkeiten bei der Bedienung des Programms während der Aufgabendurchführung.

- Wie schätzen Sie die Verständlichkeit des Programms ein? An welchen Stellen mangelt es Ihres Erachtens nach an Verständlichkeit?

- Was halten Sie hinsichtlich eines ansprechenden Designs für verbesserungswürdig?

- Nennen Sie bitte jeweils ein paar Elemente bei der Bedienung/Navigation, die Ihnen eher gut/eher weniger gut gefallen haben.

- In welchen Momenten der Bearbeitung haben Sie Freude und positive Anspannung erlebt?

- Welche Funktionen würden Sie sich von dem betrachteten Produkt für den Einsatz in Ihrer Arbeitsdomäne wünschen?

4.7.2 *formdesk* Workflowmodellierung

Bei dem Produkt *formdesk* ging es darum, dass der Proband ein Formular erstellt und dieses mit einfachen Workflow-Funktionen versieht. Die zentralen Workflowfunktion waren, dass das ausgefüllte Formular in Abhängigkeit eines gewählten Wertes einer Drop-Down-Box an unterschiedliche Empfänger geht und dass der Benutzer des Formulars nach dem Abschicken des Formulars eine Bestätigungs-E-Mail erhält. Außerdem sollte das Formular dynamisiert werden, dass eine bestimmte Drop-Down-Box nur dann eingeblendet wird, wenn zuvor bei einer Radio-Button-Gruppe ein bestimmter Wert ausgewählt worden ist. Vom Benutzer war also gefordert die unterschiedlichen E-Mail-Berichte des Workflows zu modellieren und zusätzlich die Bestätigungs-E-Mail an den Benutzer des Formulars so zu personalisieren, dass dieser mit seinem Namen begrüßt wird.

In der Beschreibung zu dem Produkt *formdesk* wird die Modellierung des umzusetzenden Workflows in BPMN vorgeschlagen. Da viele Produkte zur Workflowmodellierung diese oder eine ähnliche Art der Modellierung fordern, wurde auch dieser Schritt in die durchzuführenden Tasks aufgenommen, um zu beurteilen wie leicht oder schwer dem Probanden die Transferleistung fällt, aus der Aufgabenstellung den Workflow des Formulars zu extrahieren.

Für das Produkt *formdesk* zur einfachen Workflowmodellierung war die Aufgabenstellung die folgende:

- Erstellen Sie ein Online-Formular, mit dem ein Kunde Probleme mit seinem Informationssystem bei Ihnen melden kann.
- Das Formular enthält beschriftete einzeilige Textfelder zur Eingabe des Namens und der E-Mail-Adresse. Beide Felder sind Pflichtfelder und sind als solche gekennzeichnet.
- Das Eingabefeld für die E-Mail-Adresse akzeptiert nur E-Mail-Adressen in einem gültigen Format.
- Der Kunde muss für sein Problem eine Priorität vergeben.
- Setzt der Kunde die Priorität seiner Problemmeldung auf ‘Hoch’, erscheint eine Auswahlbox, in der er Details zu seinem Problem angeben muss. Ist die Priorität auf ‘Niedrig’, so ist die Auswahlbox für die Details nicht zu sehen.
- Das Formular enthält ein beschriftetes mehrzeiliges Texteingabefeld, in dem der Kunde seine Problembeschreibung eingeben muss. Das mehrzeilige Textfeld für die Problembeschreibung ist ein Pflichtfeld und als solches gekennzeichnet.
- Das Formular enthält einen Button zum Versenden des Formulars, welcher mit ‘Problem melden’ beschriftet ist.

- In Abhängigkeit von der gewählten Priorität wird das Formular an unterschiedliche Empfänger verschickt.
- Der Absender des Formulars bekommt nach dem Abschicken des Formulars eine E-Mail-Benachrichtigung, in der er mit Anrede und Namen begrüßt wird und die den Text 'Wir haben Ihre Problemmeldung erhalten' enthält.

Daraus ergaben sich für den Probanden die folgenden durchzuführenden Tasks:

1. Verschaffen Sie sich anhand der auf dem Desktop liegenden Datei zur Einführung in die Business Process Modeling Notation (BPMN) einen Überblick über die BPMN und deren Elemente zur Prozessmodellierung.
2. Überlegen Sie, wie der Workflow des Formulars und der entsprechenden E-Mail-Benachrichtigungen ist und welche Akteure daran beteiligt sind. Versuchen Sie, dieses in BPMN auf einem Blatt Papier oder in dem Programm OmniGraffle zu modellieren.
3. Verschaffen Sie sich einen Überblick über die Möglichkeiten der Anwendung *formdesk*, insbesondere über die Möglichkeiten von Formularen, Workflows, E-Mail-Berichten und Workflow-Berichten.
4. Loggen Sie sich mit dem Benutzernamen julia.schenk@infopark.de ein und erstellen Sie ein neues Formular mit zentriertem Layout.
5. Fügen Sie dem Formular die Überschrift 'Trouble-Ticket' hinzu.
6. Fügen Sie dem Formular beschriftete einzeilige Texteingabefelder zur Eingabe des Namens und der E-Mail-Adresse hinzu. Beide Eingabefelder sind Pflichtfelder. Die Felder haben eine Breite von 25 Zeichen und es sind maximal 25 Zeichen als Eingabe erlaubt. Das Eingabefeld für die E-Mail-Adresse erlaubt nur E-Mail-Adressen in einem gültigen Format.
7. Erstellen Sie zwei beschriftete Radio-Buttons zum Festlegen der Priorität 'Hoch' oder 'Niedrig'. Die Angabe einer Priorität ist ein Pflichtfeld und als solches gekennzeichnet. Es darf nur einer der Radio-Buttons auswählbar sein.
8. Fügen Sie neben die Radio-Buttons zur Prioritätsvergabe eine mit 'Details' beschriftete Drop-Down-Box ein, welche die Werte 'technisches Problem' und 'Softwarefehler' enthält. Es ist nur einer der beiden Werte auswählbar. Legen Sie fest, dass die Auswahlbox für die Details nur dann erscheint, wenn die Priorität 'Hoch' ausgewählt ist. Die Angabe von Details ist bei ausgewählter Priorität 'Hoch' ein Pflichtfeld.
9. Nun sind alle Felder des Formulars erstellt. Legen Sie jetzt fest, dass der Button zum Abschicken des Formulars mit 'Problem melden' beschriftet ist.

10. Legen Sie fest, dass der Benutzer des Formulars nach dem Abschicken eine E-Mail-Benachrichtigung bekommt, in der er mit seinem Namen begrüßt wird und die den Text 'Wir haben Ihre Problemmeldung erhalten' enthält. Die Benachrichtigungs-E-Mail hat den Betreff 'Vielen Dank für Ihre Meldung'.

11. Legen Sie fest, dass das Formular in Abhängigkeit von der gewählten Priorität an unterschiedliche Empfänger geht. Ein Formular mit ausgewählter hoher Priorität geht an hochprio@infopark.de, ein Formular mit niedriger Priorität geht an niedrigprio@infopark.de.

Das anschließende Interview umfasste je nach Proband die folgenden Fragen:

Als Einstieg

- Was sagen Sie spontan zu dem Produkt?
- Was sind Ihrer Meinung nach die Stärken und Schwächen des Produkts?
- Wie leicht/schwer viel Ihnen das Verständnis und die Modellierung der Aufgabenstellung in BPMN?

Fragen/Themen, die ggf. durch die Einstiegsfragen nicht betrachtet wurden:

- Nennen Sie bitte jeweils ein paar Dinge, die Ihnen an dem Programm gut/weniger gut gefallen haben.
- Beschreiben Sie bitte ihre Schwierigkeiten bei der Bedienung des Programms während der Aufgabendurchführung.
- Wie schätzen Sie die Verständlichkeit des Programms ein?
- An welchen Stellen mangelt es Ihres Erachtens nach an Verständlichkeit?
- Was halten Sie hinsichtlich eines ansprechenden Designs für verbesserungswürdig?
- Nennen Sie bitte jeweils ein paar Elemente bei der Bedienung/Navigation, die Ihnen eher gut/eher weniger gut gefallen haben.
- In welchen Momenten der Bearbeitung haben Sie Freude und positive Anspannung erlebt?
- Welche Funktionen würden Sie sich von dem betrachteten Produkt für den Einsatz in Ihrer Arbeitsdomäne wünschen?

4.7.3 *WebDSL*

Bei dem Produkt *WebDSL*, einer domänenspezifischen Sprache für dynamische Webanwendungen, ging es hauptsächlich darum, die Verständlichkeit der Formular- und Workflowmodellierung in textueller (verbaler) Notation sowie deren Akzeptanz bei den Domänenexperten zu testen. Dazu wurden den Probanden eine kurze Übersicht der Konzepte der *WebDSL* sowie einige einfache Beispiele von in *WebDSL* formulierten Formularseiten gezeigt.

In den gezeigten Beispielen werden HTML-nahe Schlüsselwörter wie

```
title, form, input, action, table, row
```

verwendet.

Die Herausforderung in diesem Test war es, dass der Benutzer anhand der Schlüsselwörter die Strukturen der Beispiele erkennt und ggf. in der Lage ist, aus den gezeigten Beispielen und der Dokumentation heraus selbst eine einfache Seite mit einem Formular und zwei Eingabefeldern in *WebDSL* abzubilden. Es wurden also nur die einfachsten Aspekte der Seiten- und Formulardefinition betrachtet. Die Konzepte der Daten- und Workflowmodellierung sowie die Definition von Prozeduren wurden gänzlich außen vor gelassen.

```
#### Wiki Page Edit
define page editPage(p : Page) {
    var newTitle : String := p.title;
    var newContent : WikiText := p.content;
    main()
    title{"Edit Page: " output(p.name)}
    define body() {
        section {
            header{"Edit Page: " output(p.name)}
            form {
                par{ input(newTitle) }
                par{ input(newContent) }
                par{ action("Save changes", savePage()) }
                action savePage() {
                    p.makeChange(newTitle, newContent, securityContext.principal)
                        ;
                    // makeChange stores patch to keep history
                    p.persist();
                    return page(p);
                }
            }
        }
    }
}

define page commentReport(p:ProgressMeeting){
    main()
    define body(){
        header{"Comment report" output(p.employee)}
        form{
            table{
                row{"Employee: " output(p.employee)}
                row{"Report: " output(p.report)}
                row{"Comments: " input(p.comments)}
            }
            action("Submit",submit())
            action submit(){
            return home();
        }
    }
}
```

Listing 4.1: *WebDSL* Beispielcode

Dem Benutzer wurden die folgenden Aufgabenstellungen vorgelegt:

- Verschaffen Sie sich anhand der Dokumente in dem auf dem Desktop liegenden Ordner 'webdsl' einen Überblick über die Konzepte der domänenspezifischen Sprache *WebDSL*, welche zur Modellierung von Webanwendungen, also auch von Seiten und Formularen, gedacht ist.
- Versuchen Sie anhand der Dokumentation und der aufgeführten Beispiele in dem Texteditor 'Textmate' [1] in *WebDSL* eine Seite zu definieren, welche wiederum ein leeres Formular enthält.
- Definieren Sie in dem Formular nun zwei Textfelder zur Eingabe eines Namens und einer E-Mail-Adresse.

Eine mögliche Lösung konnte durch Copy-Paste aus den Beispielen heraus extrahiert werden und sollte in etwa die folgenden Strukturen enthalten:

```
define page formPage() {
    var name : String;
    var email : email;
    main()
    define body() {
        section {
            form {
                par{ input(name) }
                par{ input(email) }
                action("Submit",submit())
                action submit(){
                    return home();
                }
            }
        }
    }
}
```

Listing 4.2: Einfache Formularstrukturen in *WebDSL*

In dem anschließenden Interview wurden die folgenden Fragestellungen betrachtet:

- Was sagen Sie spontan zu dieser Art der Problemmodellierung?
- Was meinen Sie, könnten die Vor- und Nachteile dieser textuellen Ausdrucksweise in Ihrer Arbeitsdomäne sein?
- Wie schätzen Sie die Verständlichkeit der formulierten Beispiele ein?

[1] siehe http://macromates.com/

5 Auswertung und Analyse der erhobenen Testdaten

Während der Durchführung der Usability-Tests entstanden Testdaten in Form von der Screen-Capture Software aufgezeichneten Video- und Audiodaten, der Audioaufzeichnung des durchgeführten Interviews und des Testplans. Diese Daten werden im Folgenden für jedes getestete Produkt ausgewertet. Um möglichst ergiebige Ergebnisse zu erhalten, ist es vorher jedoch notwendig, sich eine geeignete Granularität der zu analysierenden Punkte sowie angemessene Metriken zur Bewertung dieser Punkte zu überlegen.

Es wurde daher als sinnvoll erachtet, zur Auswertung der Testdaten die einzelnen Tasks in ihre durchgeführten Einzelaktionen, im Folgenden als **Mikrotasks** bezeichnet, aufzuschlüsseln und separat zu bewerten. Für die Auswertung der Testdurchführungen mit der Anwendung *jotform* ergaben sich die in Tabelle 5.1 aufgeführten Mikrotasks:

Task	Mikrotask	Beschreibung
1	1	Einloggen in die Anwendung jotform
2	2	ein neues, leeres Formular erstellen
	3	dem Formular den ‚default' Style zuweisen
3	4	Hinzufügen der Radio-Buttons für die Anrede
	5	Beschriftung der Radio-Buttons für die Anrede
	6	die Werte der Radio-Buttons für die Anrede setzen
	7	Vorauswahl bei den Radio-Buttons auf den Wert 'Herr' setzen
	8	Kennzeichnen der Anrede als Pflichtfeld
4	9	Hinzufügen des einzeiligen Textfeldes für den Namen
	10	Beschriften des einzeiligen Textfeldes für den Namen
	11	Hinzufügen des einzeiligen Textfeldes für den Vornamen
	12	Beschriften des einzeiligen Textfeldes für den Vornamen
	13	Kennzeichnen des einzeiligen Textfeldes für den Namen als Pflichtfeld
	14	Setzen der Breite und der max. Zeichenanzahl des Textfeldes für den Namen
	15	Setzen der Breite und der max. Zeichenanzahl des Textfeldes für den Vornamen

Task	Mikrotask	Beschreibung
5	16	Hinzufügen des einzeiligen Textfeldes für die E-Mail-Adresse
	17	Beschriften des einzeiligen Textfeldes für die E-Mail-Adresse
	18	Kennzeichnen des einzeiligen Textfeldes für die E-Mail-Adresse als Pflichtfeld
	19	Festlegen der Breite und max. Zeichenanzahl für das einzeilige Textfeld für die E-Mail-Adresse
	20	Validierung der Eingabe für das Textfeld der E-Mail-Adresse festlegen
6	21	Hinzufügen des Textbereiches für die optionale Mitteilung hinzufügen
	22	Beschriften des Textbereiches für die optionale Mitteilung
	23	Festlegen der Breite von 30 Zeichen und der Höhe von 10 Zeilen für den Textbereich für die optionale Mitteilung
	24	Eingabe der Textvorbelegung für den Textbereich für die optionale Mitteilung
7	25	Hinzufügen der Drop-Down-Box für die Veranstaltungstitel
	26	Beschriften der Drop-Down-Box für die Veranstaltungstitel
	27	Eintragen der Werte für Drop-Down-Auswahl für die Veranstaltungstitel
	28	Es soll nur ein Wert in der Drop-Down-Auswahl für die Veranstaltungstitel auswählbar sein
	29	Markieren der Drop-Down-Box für die Veranstaltungstitel als Pflichtfeld
8	30	Hinzufügen der Drop-Down-Box für die Veranstaltungstermine
	31	Beschriften der Drop-Down-Box für Termine
	32	Eintragen der Werte für die Drop-Down-Box für die Veranstaltungstitel
	33	Markieren der Drop-Down-Box für die Veranstaltungstitel als Pflichtfeld
	34	Erkennen, dass die Funktion zum Herstellen der Abhängigkeit zwischen Veranstaltung und Terminen nicht geht
9	35	Den Button zum Abschicken des Formulars mit ‘Anmelden’ beschriften
10	36	Herausfinden, dass es nicht möglich ist zwei Empfänger für die Benachrichtigungs-E-Mail mit den Formulardaten einzutragen
11	37	Bestätigungs-E-Mail an den Benutzer des Formulars aktivieren

Tabelle 5.1: *jotform*: Mikrotasks

Für das getestete Produkt *formdesk* ergaben sich für die Auswertung die in Tabelle 5.2 gezeigten Mikrotasks:

Task	Mikrotask	Beschreibung
1	1	groben Überblick über die BPMN verschaffen
2	2	Extrahieren und Modellieren des Workflows in BPMN (oder ähnlich)
3	3	Überblick über die Anwendung formdesk verschaffen
4	4	Einloggen in die Anwendung formdesk
	5	ein neues Formular erstellen
	6	ein zentriertes Layout für das neue Formular setzen
5	7	dem Formular die Überschrift ‘Trouble-Ticket’ hinzufügen
6	8	Hinzufügen des einzeiligen Texteingabefeldes für den Namen
	9	Beschriften des einzeiligen Textfeldes für den Namen
	10	Setzen der Breite und max. Zeichenanzahl (25) für das Textfeld für den Namen
	11	Texteingabefeld für den Namen als Pflichtfeld setzen
	12	Hinzufügen des einzeiligen Texteingabefeldes für die E-Mail-Adresse
	13	Beschriften des einzeiligen Texteingabefeldes für die E-Mail-Adresse
	14	Setzen der Länge für das Texteingabefeld für die E-Mail-Adresse
	15	Texteingabefeld für die E-Mail-Adresse als Pflichtfeld setzen
	16	Validieren der E-Mail-Adresse sicherstellen
7	17	Hinzufügen der Radio Buttons für die Priorität
	18	Beschriften der Radio-Buttons für die Priorität
	19	Setzen der zwei Werte der Radio-Buttons für die Priorität
	20	Angabe der Priorität als Pflichtfeld setzen
8	21	Hinzufügen der Drop-Down-Box für die Details
	22	Positionieren der Drop-Down-Box für die Details neben die Drop-Down-Box für die Priorität
	23	Beschriften der Drop-Down-Box für die Details
	24	Optionswerte der Drop-Down-Box für die Details setzen
	25	Nur einer der Werte darf auswählbar sein
	26	Angabe der Details ist Pflichtfeld
	27	Einstellen, dass die Drop-Down-Box für Details nur bei Auswahl des Wertes ‘hoch’ in der Drop-Down-Box für die Priorität erscheint
9	28	Beschriften des Buttons zum Abschicken des Formulars mit ‘Problem melden’
10	29	Bestätigungs-E-Mail an den Benutzer des Formulars aktivieren
	30	Betreff der Bestätigungs-E-Mail an den Benutzer des Formulars setzen
	31	Text in der Bestätigungs-E-Mail an den Benutzer des Formulars setzen
	32	Personalisierte Anrede in der Bestätigung-E-Mail an den Benutzer des Formulars setzen

Task	Mikrotask	Beschreibung
11	33	Workflow-Bericht erstellen, der bei gewählter Priorität 'hoch' an einen Empfänger aus dem Unternehmen geht
	34	Empfängeradresse für den Workflow-Bericht bei ausgewählter Priorität 'hoch' setzen
	35	Einstellen, dass der Workflow-Bericht auch nur bei ausgewählter Priorität 'hoch' verschickt wird
	36	Workflow-Bericht erstellen, der bei gewählter Priorität 'niedrig' an einen Empfänger aus dem Unternehmen geht
	37	Empfängeradresse für den Workflow-Bericht bei ausgewählter Priorität 'niedrig' setzen
	38	Workflow-Bericht erstellen, der bei gewählter Priorität 'niedrig' an einen Empfänger aus dem Unternehmen geht

Tabelle 5.2: *formdesk*: Mikrotasks

Bei den Tests mit der WebDSL gab es insofern keine einzelnen zu analysierenden Probleme, da die Aufgabenstellungen von keinem der Probanden selbständig ohne Hilfestellung und Erläuterungen des Testadministrators gelöst werden konnten. Daher bestehen die für dieses Modellierungskonzept erhobenen Daten aus den während der Erarbeitung der Aufgabenstellungen gemachten Äußerungen und Antworten zu den vom Testadministrator gestellten Fragen. Die Ergebnisse sind in den Audioaufzeichnungen der Testdurchführung festgehalten. Sie umfassen die Äußerungen der Probanden während der Betrachtung der Beispiele und dem Versuch, die Aufgaben durchzuführen.

5.1 Quantifizierung der aufgetretenen Usability-Probleme

Um bei einzelnen Mikrotasks den Schwierigkeitsgrad der aufgetretenen Usability-Probleme bewerten zu können, muss eine Quantifizierung der Usability-Probleme vorgenommen werden.

In [Nie94] schlagen die Autoren Nielsen und Mack die folgenden Abstufungen zur Klassifizierung des Schweregrades einen identifizierten Usability-Problems vor:

0 Dies ist kein Usability-Problem.

1 Dies ist nur ein kosmetisches Problem, das nicht behoben werden muss, solange keine zusätzliche Zeit zur Verfügung steht.

2 Dies ist ein kleines Usability-Problem - dessen Behebung erhält geringe Priorität

3 Dies ist ein großes Usability-Problem - dessen Behebung erhält große Priorität

4 Dies ist eine Usability-Katastrophe - das Problem muss unbedingt schnellstmöglich behoben werden

Da es sich bei den im Rahmen dieser Arbeit durchgeführten Usability-Tests nicht um Tests handelt, welche innerhalb eines Produkt-Entwicklungszyklus durchgeführt werden, erschien diese Abstufung der Usability-Probleme im Kontext dieser Arbeit zwar prinzipiell richtig, jedoch nicht uneingeschränkt angemessen. Im Fokus der durchgeführten Tests stand die Evaluierung der Angemessenheit verschiedener Produkte für eine bestimmte Domäne und nicht vordergründig die Priorisierung von Usability-Problemen und deren Behebung.

Aus diesem Grund wurden für die Usability-Messung eines durchgeführten Mikrotasks zweierlei Aspekte der Usability betrachtet. Zum einen wurde der Schwierigkeitsgrad der Orientierung des Benutzers auf der Oberfläche gemessen, also das Auffinden des richtigen Bedienelements zur Durchführung des Mikrotasks. Zum anderen wurde gemessen, wie problematisch die Umsetzung des Mikrotasks mit dem entsprechenden Bedienelement war. Durch diese differenzierte Betrachtungsweise der Taskdurchführung ist eine genaue Ursachenanalyse der aufgetretenen Probleme möglich.

Auch wurde innerhalb dieser beiden Usability-Aspekte jeweils eine zu [Nie94] leicht modifizierte Abstufung zur Klassifizierung des Schwierigkeitsgrades der aufgetretenen Usability-Probleme vorgenommen.

Für den Usability-Aspekt der Orientierung des Benutzers auf der Oberfläche wurde die Abstufung anhand der folgenden Indikatoren vorgenommen:

1 **Kein Problem**: Der Benutzer weiß, wo er das Bedienelement findet.

2 **Leichtes Problem**: Der Benutzer sucht das benötigte Bedienelement, findet es aber nach kurzer Suche.

3 **Mittelschweres Problem**: Der Benutzer sucht das benötigte Bedienelement, findet es nach längerer Suche

4 **Schwerwiegendes Problem**: Der Benutzer findet das benötigte Bedienelement nicht und benötigt eine Hilfestellung, um es zu finden.

Die Abstufung der Kategorisierung der Usability-Probleme bei der Umsetzung eines Mikrotasks erfolgte anhand der nachstehend aufgezeigten Kriterien.

1 **Kein Problem**: Die Aufgabe konnte selbständig ohne Probleme oder negative Aspekte umgesetzt werden.

2 **Leichtes Problem**: Die Aufgabe konnte selbständig durchgeführt werden, aber der Testperson sind negative Aspekte in der Umsetzung aufgefallen oder der Benutzer musste sich kurz orientieren.

3 **Mittelschweres Problem**: Es bestehen Verständnis- oder Orientierungsprobleme. Die Aufgabe konnte aber (nach einer gewissen Zeit) selbständig erfüllt werden.

4 **Schwerwiegendes Problem**: Die Aufgabe konnte nicht oder nur durch Eingreifen des Testleiters erfolgreich durchgeführt werden.

Zur Auswertung der Testdaten wurden für jeden Mikrotask beide Usability-Apekte betrachtet und entsprechend jeweils die aufgetretenen Usability-Probleme klassifiziert. Der folgende Ausschnitt der Auswertungstabelle illustriert die in der Auswertung erhobenen Daten für einen Probanden. Die ausführliche Tabelle der Auswertungsdaten ist im Anhang E zu finden.

formdesk Auswertung der Testdaten

formdesk Auswertung

Task	Mikrotask	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung	H.K. Bemerkungen	H.K. Min/Task
1	1	groben Überblick über die BPMN verschaffen	1	1	- schaut sich Konzepte de BPMN selbständig in pdf an	6:16
2	2	Extrahieren und Modellieren des Workflows in BPMN (oder ähnlich)	2	3	- modelliert in OmniGraffle, kennt den Umgang mit diesem Programm - schaut immer wieder in BPMN Referenz nach - sieht Problem, dass die Aufgabenstellung Informationen darüber enthält, was das Formular enthält, er aber einen Ablauf modellieren soll - erkennt Schwierigkeit der Transferleistung	16:00
3	3	Überblick über die Anwendung formdesk verschaffen	1	1		3:43
4	4	Einloggen in die Anwendung formdesk	1	1		1:54
4	5	ein neues Formular erstellen	1	2	- findet schnell Menüpunkt (unten) um neues Formular zu erstellen	
4	6	ein zentriertes Layout für das neue Formular setzen	1	4	- sieht eine Formularvorschau - Hinweis: auf „weiter“ klicken um Layout zu übernehmen	
5	7	dem Formular die Überschrift „Trouble-Ticket“ hinzufügen	1	1	- macht Dopelklick auf „Kopftext“	1:00
6	8	Hinzufügen des einzeiligen Texteingabefeldes für den Namen	1	1		4:37
6	9	Beschriften des einzeiligen Textfeldes für den Namen	1	2	- negativer Aspekt: Bezeichnung „Fragestellung“ für Titel des Feldes irritiert - weitere Optionen sind verwirrend	
6	10	Setzen der Breite und max. Zeichenanzahl (25) für das Textfeld für den Namen	1	1		
6	11	Texteingabefeld für den Namen als Pflichtfeld setzen	1	2	- Bezeichnung „Einfuhr verpflichtet“ für Pflichtfeld verwirrt	
6	12	Hinzufügen des einzeiligen Texteingabefeldes für die E-Mail-Adresse	1	1		
6	13	Beschriften des einzeiligen Texteingabefeldes für die E-Mail Adresse	1	1		
6	14	Setzen der Länge für das Texteingabefeld für die E-Mail Adresse	1	1		
6	15	Texteingabefeld für die E-Mail-Adresse als Pflichtfeld setzen	1	1		
6	16	Validierung der E-Mail-Adresse sicherstellen	2	2	- hat zunächst vergessen Validierung einzustellen, sucht im oberen Menü die Möglichkeit in Bearbeitungsmodus des Feldes zurück zu kommen - sieht dort die Option „E-Mail“, vermutet, dass es sich hierbei um die Validierung handelt.	
7	17	Hinzufügen der Radio Buttons für die Priorität	1	1		1:35
7	18	Beschriftung der Radio-Buttons für die Priorität	1	1		
7	19	Setzen der zwei Werte der Radio-Buttons für die Priorität	1	2	- liest Hinweistext des Programms um zu wissen in welcher Form die Optionswerte einzugeben sind	
7	20	Angabe der Priorität als Pflichtfeld setzen	1	1		
8	21	Hinzufügen der Drop-Down-Box für die Details	1	1		9:08
8	22	Positionierung der Drop-Down-Box für die Deatails neben die Drop-Down-Box für die Priorität	1	2	- findet im linken Menü den Punkt „Ausrichten“ - gewünschter Auswahlpunkt ist nicht auswählbar, erst nachdem voranstehender Menüpunkt aktiviert wurde --> unklar warum	
8	23	Beschriftung der Drop-Down Box für die Details	1	1		
8	24	Optionswerte der Drop-Down Box für die Details setzen	1	1		
8	25	Nur einer der Werte darf auswählbar sein	1	1		
8	26	Angabe der Details ist Pflichtfeld	1	1		
8	27	Einstellen, dass die Drop-Down Box für Details nur bei Auswahl des Wertes „hoch“ in der Drop-Down-Box für die Priorität erscheint	4	4	- Hinweis: obere Registerkarten repräsentieren einzelne Schritte - Hinweis: Abhängigkeit ist Eigenschaft des Felds - Hinweis: Es ist eine Voraussetzung für dieses Feld, daraufhin orientiert sich Benutzer im linken Menü - Probleme bei Begriffen zur Bedingungsdefinition, Hilfestellung notwendig - Lösch-Icon als solches nicht erkennbar	
9	28	Beschriften des Buttons zum Abschicken des Formulars mit „Problem melden“	3	2	- sucht Button in Liste der Elemente die hinzugefügt werden können, findet ihn dort nicht, klickt auf ‚weiter‘	
10	29	Bestätigungs-E-Mail an den Ausfüller des Formulars aktivieren	3	4	- sucht unter Einstellungen Möglichkeit Bestätigungsemail einzustellen - schaut im linken Menü, findet dort die Option „Email/ Workflow“ , findet sich innerhalb diese Menüs aber nicht zurecht - Hinweis: Unterschied Workflowbericht (intern) und E-Mail Bericht (an Kunde)	9:24
10	30	Betreff der Bestätigungs-E-Mail an den Ausfüller setzen	4	1	- Bei Berichterstellung „Allgemein“: Beschreibung=Betreff?, sucht Stelle wo Inhalt des Berichts zu beschreiben ist - Hinweis: im Linken Menü schauen für Betreff etc. setzen (Menüpunkt „E-Mail Bericht“)	
10	31	Text in der Bestätigungs-E-Mail an den Ausfüller setzen	1	1		

Abbildung 5.1 zeigt für das Produkt *jotform* pro Proband die grafische Auswertung der aufgetretenen Usability-Probleme für den Aspekt der Umsetzung der einzelnen Mikrotasks. Im Anhang E finden sich die weiteren grafischen Auswertungen der beiden evaluierten Produkte.

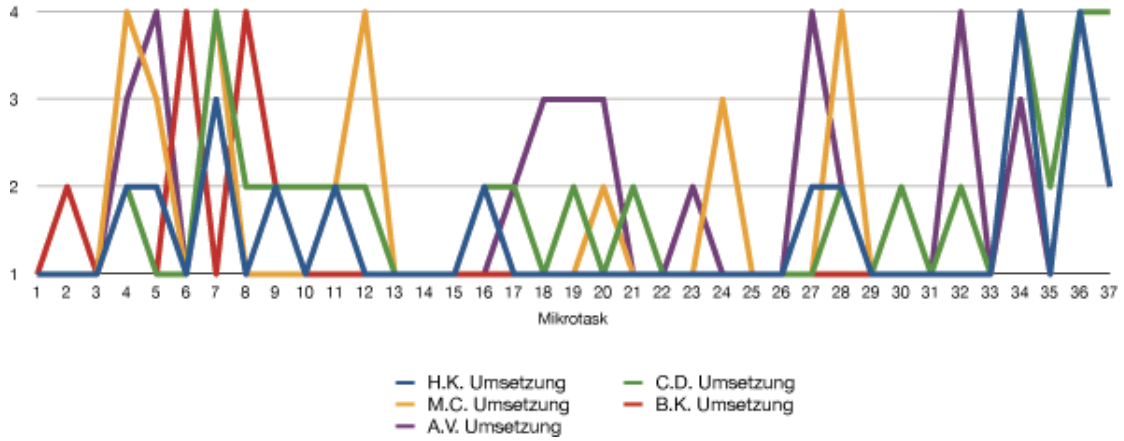


Abbildung 5.1: *jotform*: Usability-Probleme der einzelnen Probanden für den Aspekt der Umsetzung bei den einzelnen Mikrotasks

5.2 Mittelwert der aufgetretenen Usability-Probleme pro Mikrotask

Um eine Übersicht der problembehafteten Mikrotasks zu bekommen, wurde pro Mikrotask aus den Problemgraden beider Usability-Aspekte und aller Probanden der Mittelwert gebildet. Für die Anwendung *jotform* ergibt sich damit der in Abbildung 5.2 gezeigte Verlauf.

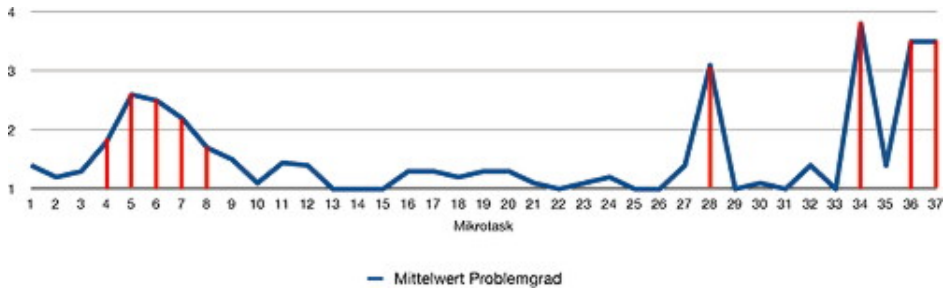


Abbildung 5.2: *jotform*: Mittelwert der aufgetretenen Usability-Probleme aller Probanden pro Mikrotask

Der Mittelwert von Mikrotasks, welcher im überdurchschnittlich hohen Bereich liegt, ist durch eine rote Linie hervorgehoben. Diese Mikrotasks werden als kritisch betrachtet und daher bezüglich ihrer Usability-Probleme im Weiteren genauer untersucht werden.

Tabelle 5.3 zeigt eine Zusammenfassung der Mikrotasks nach ihren zentralen Aufgabenaspekten. Betrachtet man den in Abbildung 5.2 gezeigten Verlauf der Usability-Probleme im Zusammenhang mit Tabelle 5.3, so wird deutlich, dass bei dem Hinzufügen und Bearbeiten von Formularelementen lediglich beim ersten Mal Probleme und Unsicherheiten bei den Probanden auftraten. Es zeigt sich eine deutliche Lernkurve, da bei weiteren Tasks der gleichen Art im Mittel keine größeren Probleme mehr auftraten. Erst wieder bei neuartigen Tasks, welche fehlende oder sehr versteckte Funktionen umfassten, gab es sehr große Probleme bei den Benutzern.

zentraler Aspekt	erster durchgeführter Mikrotask zu diesem Aspekt	wiederholende Mikrotasks zu diesem Aspekt
einloggen	1	-
Wizard Benutzung: neues Formular	2	3
ein Formularelement hinzufügen	4	9, 11, 16, 21, 25, 30
ein Formularelement beschriften	5	10, 12, 17, 22, 26, 31
Werte eines Formularelements setzen	6	27, 32
Vorauswahl eines Wertes eines Formularelements setzen	7	-
Setzen eines Formularelements als Pflichtfeld	8	13, 18, 29, 33
Breite eines Formularelements setzen	14	15, 19, 23
Aktivieren der E-Mail Validierung	20	-
Eingeben einer Textvorbelegung	24	-
Es soll bei einer Drop-Down-Auswahl nur ein Wert auswählbar sein	28	-
Erkennen einer fehlenden Funktion	34	36
Beschriften des Submit-Buttons	35	-
E-Mail an den Benutzer des Formulars aktivieren	37	-

Tabelle 5.3: *jotform*: Zentrale Aspekte der Mikrotask

Für die Anwendung *formdesk* ergibt sich der in Abbildung 5.3 gezeigte Verlauf der Usability-Probleme bei den durchgeführten Mikrotasks:

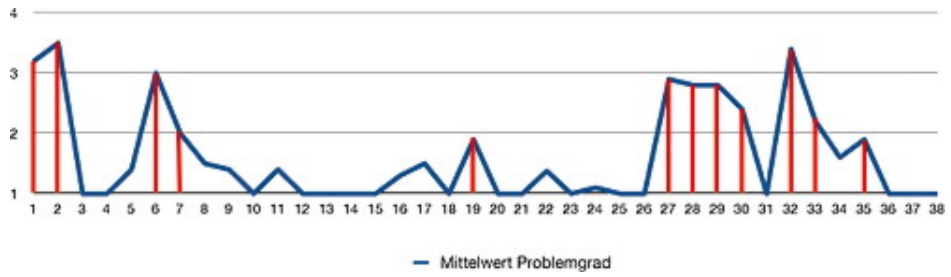


Abbildung 5.3: *formdesk*: Mittelwert der aufgetretenen Usability-Probleme aller Probanden pro Mikrotask

Auch ist es sinnvoll, diesen Verlauf wieder unter Bezugnahme der in Tabelle 5.4 nach Aufgabenaspekten aufgegliederten Mikrotasks zu betrachten.

Dabei wird deutlich, dass alle Mikrotasks, die die Modellierung des Workflows in der BPMN sowie die Navigation durch den Gesamtprozess betreffen, bei den Benutzern immer wieder zu gravierenden Problemen führten. Das Einfügen und Bearbeiten der Eigenschaften von Formularelementen bereitete den Benutzern keine bemerkenswerten Probleme, sofern dieses nicht eine Navigation innerhalb des Bearbeitungsmodus' der Formularelemente beinhaltete, sondern nur Optionen aus der Startansicht des Bearbeitungsmodus umfasste. Sobald jedoch Attribute oder Abhängigkeiten gesetzt werden sollten, welche ein Verständnis des Oberflächenkonzepts innerhalb des Bearbeitungsmodus' von Formularelementen, und damit eine Orientierung in der linken Navigation des Bearbeitungsmodus' erforderten, kam es zu massiven Orientierungsproblemen der Benutzer.

Die gleichen Probleme gab es auch in der Bearbeitung von E-Mail- und Workflow-Berichten. Auch hier war die Durchführung der Mikrotasks, bei denen eine Orientierung innerhalb des Bearbeitungsmodus von Berichten notwendig war, stark problematisch bzw. nicht ohne Hilfestellung durchführbar. Beim Erstellen von Berichten waren bei jeder neuen Berichtsart Hilfestellungen durch den Testadministrator notwendig. Berichte der selben Art konnten dann aber ohne Probleme erstellt werden. Die Verwendung von Formularvariablen war für die Benutzer ebenfalls nicht verständlich.

zentraler Aspekt	erster durchgeführter Mikrotask zu diesem Aspekt	wiederholende Mikrotasks zu diesem Aspekt
BPMN / Workflow	1	2
Überblick und einloggen	3	4
neues Formular erstellen	5	-
Navigation durch den Modellierungsprozess	6	28
Sonstige Eigenschaften von Formularelementen setzen	7	-
ein Formularelement hinzufügen	8	12, 17, 21
ein Formularelement beschriften	9	13, 18, 23
Breite eines Formularelements setzen	10	14
Setzen eines Formularelements als Pflichtfeld	11	15, 20, 26
Aktivieren der E-Mail Validierung	16	-
Werte eines Formularelements setzen	19	24
Positionierung eines Formularelements	22	-
Es soll bei einer Drop-Down-Auswahl nur ein Wert auswählbar sein	25	-
Abhängigkeit von Formularelementen	27	-
E-Mail-Bericht an den Benutzer des Formulars aktivieren	29	-
Inhalte des E-Mail-Berichts setzen	30	31
Konzept der Formularvariablen verstehen	32	-
Berichtsart für Workflow-Bericht erkennen und diesen erstellen	33	36
Eigenschaften eines Workflow-Berichts bearbeiten	34	35, 37, 38

Tabelle 5.4: *formdesk*: Zentrale Aspekte der Mikrotasks

5.3 Analyse der aufgetretenen Usability-Probleme

Im Folgenden werden die aufgetretenen Usability-Probleme für jede getestete Anwendung untersucht und hinsichtlich ihrer Ursachen analysiert. Basierend darauf soll dann für jedes Produkt herausgestellt werden, bei welchen Punkten es hauptsächlich zu Problemen in der Bedienung kommt.

In [Eicd] empfiehlt der Autor zur Auswertung von Usability-Tests eine Liste der aufgetretenen Probleme zu erstellen. Des Weiteren empfiehlt er, zur besseren Analyse die Probleme nach Ähnlichkeit oder nach Art bzw. Größe ihres Einflusses zu gruppieren.

Um die bei den einzelnen Mikrotasks aufgetretenen Probleme innerhalb der Bedienungsaspekte genauer analysieren zu können, wurden die aufgetretenen Usability-Probleme zunächst hinsichtlich der Art des aufgetretenen Problems kategorisiert. Im Kontext der in dieser Arbeit durchgeführten Usability-Tests haben sich die folgenden Problemkategorien herausgestellt:

Orientierunglosigkeit im Modellierungsprozess	(OrientLosModProz)
Suche nach Bedienelement	(SuchBedElem)
Unsicherheit ob richtiges Bedienelement gewählt	(UnsicherhRichtBedElem)
Unsicherheit über Richtigkeit der durchgeführten Aktion	(UnsicherhRichtAktion)
Verunsicherung durch zahlreiche Auswahlmöglichkeiten	(VerunsZahlrAusw)
Probleme in der Bedienung des Layouts	(ProbBedLay)
(Bedien-) Konzept unklar	(KonzUnklar)

Da Usability-Probleme der gleichen Kategorie unterschiedliche Ursachen haben können, musste eine differenzierte Analyse und Ursachenzuordnung für jedes einzelne Problem möglich sein. Dazu wurden die verschiedenen möglichen Ursachen für Usability-Probleme in diesem Kontext identifiziert:

1. keine/schlechte Führung durch den Gesamtprozess
2. Konzept/Aufbau der Software unklar
3. fehlendes Feedback
4. fehlende Information
5. schlechte Strukturierung der Oberfläche
6. schlechte Bedienbarkeit
7. nicht aussagekräftige Bezeichnungen/Icons
8. Überforderung durch Komplexität
9. inkonsistentes Bedienungskonzept

5.3.1 Analyse für den *jotform* Formulargenerator

Das Bilden der Mittelwerte in Abschnitt 5.2 zeigt in Abbildung 5.2, dass für das Produkt *jotform* Formulargenerator die folgenden Tasks im kritischen Bereich der Usability-Probleme liegen:

4 Hinzufügen der Radio-Buttons für die Anrede

5 Beschriftung der Radio-Buttons für die Anrede

6 Die Werte der Radio-Buttons für die Anrede setzen

7 Vorauswahl bei den Radio-Buttons auf den Wert 'Herr' setzen

8 Kennzeichnung der Anrede als Pflichtfeld

28 Es soll nur ein Wert in der Drop-Down-Auswahl für die Veranstaltungstitel auswählbar sein.

34 Erkennen, dass die Funktion zum Herstellen der Abhängigkeit zwischen Veranstaltung und Terminen nicht geht.

36 Herausfinden, dass es nicht möglich ist, zwei Empfänger für die Benachrichtigungs-E-Mail mit den Formulardaten einzutragen.

37 Die Bestätigungs-E-Mail an den Benutzer des Formulars aktivieren

Für die als kritisch identifizierten Mikrotasks sollen nun die aufgetretenen Probleme genauer untersucht werden. Zunächst werden in Tabelle 5.5 für jeden kritischen Mikrotask alle aufgetretenen Probleme sowie deren Zuordnung zu einer Problemkategorie und Ursache aufgezeigt.

Mikrotask	Problem	Häufigkeit	Problemkategorie	Ursache
4	Proband hat Probleme mit der Drop-Zone	3	ProbBedLay	6
	Proband hätte nicht gewusst, was Radio-Buttons sind	1	UnsicherhRichtBedElem	7
5	keine Markierung wie Rahmen etc., wo die Beschriftung im Formularelement zu setzen ist	1	ProbBedLay	6
	Proband versucht, über das Submit-Button Element des Formulars Eingaben zu bestätigen	1	KonzUnklar	2
	Proband versucht, ein Text-Box-Element als Beschriftung hinzuzufügen	1	KonzUnklar	2
	Irritiert über 'Click me to edit'	1	KonzUnklar	7
	Proband weiß nicht, wo Beschriftung zu setzen ist	3	SuchBedElem	2
	Das Wort 'Label' wird im Formularelement nicht angezeigt. Unklarheit, was die Option Label im Eigenschaften-Inspektor ist	1	UnsicherhRichtBedElem	9
6	Problem: flackernder Eigenschaften-Inspektor	1	ProbBedLay	6
	Proband schafft es nicht, die vorgegebene 3. Option bei den Werten für die Radio-Buttons zu löschen	1	ProbBedLay	4
	Proband findet (erkennt) Eigenschaften-Inspektor nicht als solchen	1	KonzUnklar	5
	Proband weiß nicht, wo die Werte für die Radio-Buttons zu setzen sind	4	SuchBedElem	2
7	Proband kommt nicht mit Bedienung des Eigenschaften-Inspektors zurecht	1	ProbBedLay	6
	Problem: flackernder Eigenschaften-Inspektor	1	ProbBedLay	6
	Proband weiß nicht, wie der vorzuselektierende Wert einzutragen ist	4	KonzUnklar	4
	Proband ist unsicher über Verwendung des vorhandenen Submit-Button-Elements im Formular	1	KonzUnklar	2
	Proband schafft es trotz Hinweis nicht, die Vorauswahl zu setzen	1	KonzUnklar	4
	Unklarheit über Verwendung des Submit-Buttons	1	KonzUnklar	2
	dem Probanden fehlt das Feedback darüber, was als Vorauswahl gesetzt ist	5	UnsicherhRichtAktion	3
8	Proband bestätigt Eingaben über das Submit-Button-Element des Formulars	1	KonzUnklar	2
	Proband findet die Option nicht, um das Formularelement als Pflichtfeld zu setzen	1	SuchBedElem	2
	Proband sucht Text, um Sternchen zu beschreiben	1	SuchBedElem	4
28	Proband findet nicht Option zum Einstellen, dass nur eine Veranstaltung auswählbar ist	5	SuchBedElem	4
	Proband probiert die Option 'size' aus, ist sich aber nicht sicher, ob dort richtige Einstellung	2	UnsicherhRichtBedElem	7
	Unklarheit beim Probanden über die Option 'special'	3	VerunsZahlrAusw	7
34	Für den Probanden ist nicht erkennbar, dass diese Funktion nicht umsetzbar ist	5	OrientLosModProz	4
	Proband sucht nach Verknüpfungsmöglichkeit der Veranstaltungen und deren Termine	5	SuchBedElem	4

Mikrotask	Problem	Häufigkeit	Problemkategorie	Ursache
36	Proband versucht über Anklicken des Submit-Button Elements weiter zu kommen	1	KonzUnklar	2
	Proband findet die Eigenschaften des gesamten Formulars nicht	1	KonzUnklar	5
	Proband erkennt nicht, dass es nicht möglich ist, eine zweite E-Mail-Adresse einzutragen.	4	OrientLosModProz	4
	Proband weiß nicht, wo die die Stelle ist, um die zweite E-Mail-Adresse einzutragen	3	SuchBedElem	5
	Proband probiert Wizard-Button, klickt durch und probiert an richtige Stelle zu kommen	1	SuchBedElem	1
	Proband ist unsicher, ob und in welcher Form eine zweite E-Mail einzutragen ist	4	UnsicherhRichtAktion	4
	Proband vermisst Feedback über Eingaben, würde gerne testen	3	UnsicherhRichtAktion	3
37	Proband hat Schwierigkeiten, im Wizard die Stelle zu finden, um die Bestätigungs-E-Mail an den Benutzer des Formulars zu aktivieren	4	OrientLosModProz	1
	Proband sucht wieder in den Eigenschaften des gesamten Formulars	4	SuchBedElem	9
	Proband weiß nicht, dass er den Wizard nutzen muss, um die Bestätigungs-E-Mail an den Benutzer des Formulars zu aktivieren	4	SuchBedElem	1
	Proband probiert den Wizard, ist aber verunsichert und schließt diesen wieder	1	UnsicherhRichtAktion	1
	Proband verwendet falsche Option in den Formular-Eigenschaften	1	VerunsZahlrAusw	7
	Proband ist irritiert über Bedeutung der 'Reply-To' Option innerhalb des Wizards	1	VerunsZahlrAusw	7

Tabelle 5.5: *jotform*: Aufgetretene Usability-Probleme

5.3.1.1 Anzahl der Usability-Probleme pro Mikrotask

Mithilfe von Tabelle 5.5 wird nun für jeden kritischen Mikrotask analysiert, wie viele Usability-Probleme welcher Kategorie auftauchten.

Damit ergibt sich die in Tabelle 5.6 gezeigte Verteilung der Usability-Probleme auf die einzelnen Problemkategorien und es ist anhand der Summe der aufgetretenen Probleme pro Mikrotask zu erkennen, welche kritischen Mikrotasks in ihrer Durchführung am meisten Probleme bereiteten. Dabei wird berücksichtigt, wie oft ein Problem während der Testdurchführungemn bei den Probanden auftrat. Gleichzeitig ist dabei erkennbar, in welchen Problemkategorien diese Usability-Probleme auftraten.

Mikrotask	OrientLosModProz	SuchBedElem	UnsicherhRichtBedElem	UnsicherhRichtAktion	VerunsZahlrAusw	ProbBedLay	KonzUnklar	Summe
4	0	0	1	0	0	3	0	4
5	0	3	1	0	0	1	3	8
6	0	4	0	0	0	2	1	7
7	0	0	0	5	0	2	7	14
8	0	2	0	0	0	0	1	3
28	0	5	2	0	3	0	0	10
34	5	5	0	0	0	0	0	10
36	4	4	0	7	0	0	2	17
37	4	8	0	1	2	0	0	15
Summe	13	31	4	13	5	8	14	88

Tabelle 5.6: *jotform*: Aufgetretene Usability-Probleme pro Mikrotask, gruppiert nach Problemkategorie

Tabelle 5.6 ist zu entnehmen, dass insgesamt die Durchführungen der Mikrotasks 7, 36 und 37 am stärksten problembehaftet waren. Anhand von Tabelle 5.6 lassen sich des Weiteren die folgenden Aussagen hinsichtlich der aufgetretenen Usabilty-Probleme bei dem Produkt *jotform* treffen:

I Am häufigsten traten Probleme bei der Durchführung von Mikrotask 36 auf, bei dem die Probanden herausfinden mussten, dass es nicht möglich ist, zwei Empfänger für die Benachrichtigungs-E-Mail mit den Formulardaten einzutragen. Besonders häufig waren sich die Probanden unsicher, ob die von ihnen durchgeführte Aktion richtig war, da sie nicht erkennen konnten, ob und in welcher Form es möglich ist, einen zweiten Empfänger für die Formulardaten einzugeben. Auch erhielt der Proband kein Feedback darüber, ob die von ihm eingegebenen Daten korrekt sind und übernommen wurden.

II Ebenfalls gab es eine gravierende Anzahl von Problemen bei Mikrotask 37, dem Aktivieren der Bestätigungs-E-Mail an den Benutzer des Formulars. Dort haben die Probanden analog zu dem Menüpunkt zum Einstellen des Empfängers für die Banchrichtigungs-E-Mail, die Stelle zum Aktivieren der Bestätigungs-E-Mail an den Benutzer des Formulars in den Eigenschaften des Formulars gesucht. An dieser Stelle war das Bedienungskonzept jedoch nicht konsistent und den Probanden war nicht klar, dass für die Durchführung dieser Aufgabe die Benutzung des Wizards notwendig war. Das Produkt hat an dieser Stelle keine Information darüber geliefert, dass diese Option vom System implizit aktiviert ist und ob/wo diese Option ggf. änderbar ist. Nachdem entsprechende Hinweise vom Testadministrator gegeben wurden, hatten die Probanden Probleme, mit der Führung des Wizards zurecht zu kommen und die entsprechende Stelle zur Aktivierung der Bestätigungs-E-Mail an den Benutzer des Formulars zu identifizieren.

III Bei Mikrotask 7, dem Setzen der Vorauswahl bei den Radio-Buttons auf den Wert ‘Herr’, war den Probanden nicht klar, nach welchem Konzept der vorzuselektierende Wert für die Radio-Button-Gruppe anzugeben ist. Auch waren die Probanden häufig unsicher, ob die von ihnen gesetzte Option richtig war. Das ist in diesem Falle darauf zurück zu führen, dass die Option des vorselektierten Wertes nicht in der Vorschau des Formulars angezeigt wurde, sie dieses aber erwarteten.

IV Gleichermaßen viele Probleme traten bei den Mikrotasks 28 und 34 auf. Bei Mikrotask 28 ging es darum, dass der Proband versuchte, eine implizit vom System vorgenommene Einstellung zu setzen. Da vom System keine Information darüber gegeben wurde, dass diese Option implizit gesetzt ist und somit auch keine Option verfügbar war die dem Probanden ermöglichte, diese Einstellung zu ändern, suchten die Probanden entsprechend häufig nach dieser Option. Dabei betrachteten sie andere verfügbare Optionen, bei denen sie sich aber nicht sicher waren, ob diese die richtigen Punkte sind um die Einstellung vorzunehmen.

V Task 34 umfasste eine Funktion, die im System nicht verfügbar war. Den Probanden wurde dieses vom System aus nicht deutlich gemacht, und entsprechend oft suchten die Probanden nach der Funktionalität.

5.3.1.2 Gruppierung der Usability-Probleme nach Problemkategorie

Nachdem die aufgetretenen Probleme in Kategorien eingeordnet und ihre Ursächlichkeiten untersucht wurden, ist es nun interessant, Aussagen darüber herzuleiten, welche Bereiche der Usability in einem Produkt problematisch sind. Dieses entspricht der Auswertung der letzten Zeile in Tabelle 5.5.

Dazu wurden die aufgetretenen Usability-Probleme nach Kategorien zusammen gefasst und es ergibt sich die in Abbildung 5.4 gezeigte Verteilung.

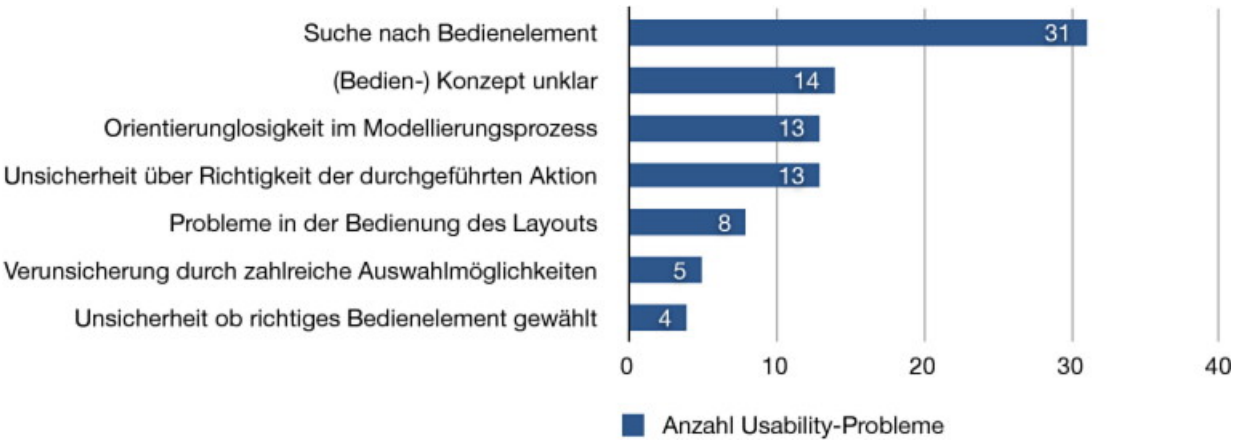


Abbildung 5.4: *jotform*: Summe der aufgetretenen Usability-Probleme pro Problemkategorie

Der größte Anteil der Probleme trat in der Problemkategorie 'Suche nach Bedienelement' auf. In diese Kategorie fallen die Mikrotasks 5, 6, 8, 28, 34, 36 und 37. Neben denen unter I, II, IV und V erläuterten Mikrotasks 28, 34, 36 und 37 liefern hier die Mikrotasks 5, 6 und 8 einen weiteren Teil der Probleme dieser Kategorie. Diese drei Mikrotasks sind jeweils die ersten durchzuführenden Mikrotasks eines bestimmten Aufgaben-Aspekts, der die

Bearbeitung bestimmter Eigenschaften von Formularelementen umfasst. Damit ist deren im Vergleich zu weiteren Mikrotasks des selben Aufgaben-Aspekts relativ hoher Anteil im Auffinden von entsprechenden Bedienelementen zu erklären. Die Probanden hatten zunächst Probleme wahrzunehmen, dass die Bearbeitung von Optionen und Werten der Formularelemente im Eigenschaften-Inspektor durchzuführen ist. Das ist darauf zurück zu führen, dass an dieser Stelle den Probanden der Aufbau und das Bedienungskonzept der Software noch nicht klar war.

Genau dieser Aspekt führt zur Problemkategorie '(Bedienungs-) Konzept unklar', in der ein weiterer sehr großer Teil der Probleme zu finden ist. Mikrotask-übergreifend bestand an vielen Punkten Unklarheit darüber, wie die vom Probanden getätigten Eingaben bei der Bearbeitung von Elementen im System übernommen werden. Dies zeigte sich daduch, dass die Probanden das im Formular enthaltene Submit-Button-Formularelement zur Bestätigung ihrer Eingaben nutzten. Auch war es den Probanden wie unter III bereits erwähnt, nicht ersichtlich, dass um bei einer Radio-Button-Gruppe einen vorselektierten Wert zu setzen, nicht, wie bei anderen Optionen, die Option selected auf yes oder no zu setzen ist, sondern dass an dieser Stelle der Wert des vorzuselektierenden Wertes einzugeben ist. An dieser Stelle wäre nach Angaben der Probanden eine Drop-Down-Box mit den möglichen Werten der Radio-Buttons verständlicher gewesen. Des Weiteren fiel es den Probanden teilweise schwer zu erkennen, dass es zum einen die Eigenschaften der einzelnen Elemente und zum anderen bestimmte Eigenschaften gibt, die dem gesamten Formular zugeordnet sind. Auch war es nicht klar, dass der Eigenschaften-Inspektor die jeweiligen Eigenschaften anzeigt.

Die Probleme der Kategorie 'Orientierungslosigkeit im Modellierungsprozess' wurden durch die Mikrotasks 34, 36 und 37 aufgebaut und sind wie unter II, IV und V bereits näher beschrieben, an den Stellen entstanden, an denen das System dem Probanden Informationen über implizite Einstellungen nicht erkenntlich und nur sehr schwer zugänglich gemacht hat.

Ebenfalls ein sehr großer Teil der Probleme bestand in der Suche nach Bedienelementen zur Durchführung der Tasks. Die Probleme in dieser Kategorie sind hauptsächlich darauf zurück zu führen, dass den Probanden nicht ersichtlich war, dass die Bearbeitung der Elemente direkt im Eigenschaften-Inspektor durchzuführen ist. Außerdem fehlte ihnen teilweise die Information, wo eine bestimmte Einstellung vorzunehmen ist.

Die Probleme aus der Kategorie 'Unsicherheit über Richtigkeit der durchgeführten Aktion' sind in den Mikrotasks 7, 36 und 37 entstanden, wobei die gravierendsten Verunsicherungen des Probanden im unter III beschriebenen Mikrotask 7 auftraten, bei dem das visuelle Feedback der vom Probanden gesetzten Option (Vorauswahl eines Wertes bei einer Radio-Button-Gruppe) fehlte.Der Proband konnte somit nicht erkennen, ob er die Einstellung richtig gesetzt hatte. Auch beim Eintragen einer weiteren E-Mail-Adresse während der Durchführung von Task 36 fehlte dem Probanden jegliches Feedback vom System bzgl. der (formalen) Korrektheit seiner eingegebenen Daten. Es gab keine Möglichkeit für den Probanden dieses anderweitig zu testen. Wie bereits unter II erwähnt, war der Wizard nicht so gestaltet, dass

er dem Probanden einen Überblick darüber gab, an welcher Stelle des Modellierungsprozesses er sich gerade befindet, welche Einstellungen an dieser Stelle optional oder verpflichtend vorzunehmen sind und welche Schritte noch folgen.

Probleme in der Bedienung des Layouts traten hauptsächlich Mikrotask-übergreifend aufgrund des schlecht funktionierenden Drag-and-Drop Mechanismus bzw. der sehr schlecht gestalteten Drop-Zone auf. Die Probanden konnten ihre Formularelemente nicht an den gewünschten Stellen im Formular ablegen, sondern mussten diese oft an einer anderen Position 'fallen lassen' und danach die Elemente in ihrer Reihenfolge verschieben. Dieses und das dem Probanden nicht nachvollziehbar flackernde Öffnen und Schliessen des Eigenschaften-Inspektors führte bei allen Probanden einen sehr hohen 'Nerv-Faktor'.

Eine Verunsicherung der Probanden durch verschiedene Auswahlmöglichkeiten trat in den beiden bereits mehrfach genannten Mikrotasks 28 und 37 auf. Es war ausnahmslos darauf zurück zu führen, dass verschiedene Optionen keine aussagekräftigen Bezeichnungen hatten. Dem Probanden wurde deshalb nicht die Funktion der jeweiligen Optionen deutlich. Dadurch war es dem Probanden nicht möglich, die für die Durchführung seiner Aufgabe richtigen Optionen zu identifizieren. Das war auch nahezu immer die Problemursache der wenigen in der Kategorie 'Unsicherheit ob richtiges Bedienelement gewählt' aufgetretenen Probleme.

5.3.1.3 Auswertung der Interviews

Bei den im Anschluss an die Testdurchführung durchgeführten Interviews ging es darum, von dem Probanden während der Testdurchführung empfundenen Eindrücke bzgl. des Produkts zu reflektieren. Alle Probanden empfanden die Idee des Drag-And-Drop Bedienungskonzeptes als sehr positiv und angenehm, bemängelten jedoch gleichermaßen die schlechte Umsetzung in dem Produkt, welche zu einem inakzeptablen Bedienungskomfort führte.

Nachdem die Probanden anfänglich Prinzip des Eigenschaften-Inspektors etwas problematisch fanden, also der Anzeige und Bearbeitung der Eigenschaften der Formular-Elemente, empfanden sie es danach jedoch als recht übersichtlich und intuitiv. Das Produkt setze auf gängige Bedienungskonzepte, die Oberfläche wurde als gut strukturiert empfunden, so dass Funktionen überwiegend gut zu finden waren und man jederzeit die Formularvorschau und die verfügbaren Funktionen im Überblick hatte. Jedoch war für die Probanden nicht nachvollziehbar, wann bzw. wodurch die Eigenschaften eines Formularelements und wann die des gesamten Formulars angezeigt werden. Dieses und das für die Probanden nicht nachvollziehbare flackernde Öffnen und Schliessen des Eigenschaften-Inspektors empfanden alle Probanden als störend.

Teilweise war die Bedienung für die Probanden nicht konsistent. Bestimmte Eingaben mussten innerhalb des Eigenschaften-Inspektors über einen Button bestätigt werden, andere wurden nach der Eingabe implizit vom System übernommen. Außerdem wurde das Konzept des

Eigenschaften-Inspektors nicht konsequent eingehalten, da manche Eingaben auch direkt in der Formular-Vorschau bearbeitet werden konnten. Den Probanden war die Bezeichnung der Bedienelemente zu fachsprachlich und zu wenig bezeichnend, so dass ihnen nicht klar war, welche Funktion sich hinter dem Bedienelement verbirgt.

Prinzipiell empfanden die Probanden das Design des Produkts als ansprechend und das Erstellen eines Formulars als einfach. Beispielsweise empfanden sie beim Setzen von Optionen eine Einschränkung der möglichen Werte durch eine Drop-Down-Liste als sehr hilfreich und fehlervermeidend. Andererseits bemängelten sie das fehlende Feedback in der Formular-Vorschau zu bestimmten Optionen. Dadurch fühlten sie sich stark verunsichert bzgl. der Korrektheit ihrer durchgeführten Aktion. Als Verbesserungpotential wurden bessere Feedbackmeldungen zu den vom Benutzer durchgeführten Aktionen und erläuternde Texte bei Mouse-Over der Funktionen genannt.

Nach Meinung der Probanden fehlen dem Produkt grundlegende Funktionen für ihre Problemdomäne. Dazu zählen die Möglichkeit, das Formular an mehrere Empfänger oder in Abhängigkeit der eingegebenen Werte an unterschiedliche Empfänger zu senden. Ebenfalls gewünscht wurde die Gestaltung des Inhalts der Bestätigungs-E-Mail an Benutzer des Formulars. Auch würden die Probanden gerne erweiterte Möglichkeiten zur Bearbeitung des Formular-Designs haben oder sich beispielsweise die Formulardaten gesammelt als csv-Datei zuschicken lassen.

5.3.2 Analyse für das *formdesk* Formular-Management-System

Auch für das Formular-Management-System *formdesk* wurden im Abschnitt 5.2 die Mittelwerte der Usability-Probleme aller Probanden für jeden durchgeführten Mikrotask gebildet. Abbildung 5.3 zeigt, dass für das Produkt *formdesk* die folgenden Tasks im kritischen Bereich der Usability liegen:

1 groben Überblick über die BPMN verschaffen

2 Extrahieren und Modellieren des Workflows in BPMN (oder ähnlich)

6 ein zentriertes Layout für das neue Formular setzen

7 dem Formular die Überschrift 'Trouble-Ticket' hinzufügen

19 Setzen der zwei Werte der Radio-Buttons für die Priorität

27 Einstellen, dass die Drop-Down-Box für Details nur bei Auswahl des Wertes 'hoch' in der Drop-Down-Box für die Priorität erscheint

28 Beschriften des Buttons zum Abschicken des Formulars mit 'Problem melden'

29 Bestätigungs-E-Mail an den Benutzer des Formulars aktivieren

30 Betreff der Bestätigungs-E-Mail an den Benutzer des Formulars setzen

32 Personalisierte Anrede in der Bestätigung-E-Mail an den Benutzer des Formulars setzen

33 Workflow-Bericht erstellen, der bei gewählter Priorität 'hoch' an einen Empfänger aus dem Unternehmen geht

35 Einstellen, dass der Workflow-Bericht auch nur bei ausgewählter Priorität 'hoch' verschickt wird

Wie auch für das Produkt *jotform* erfolgt für das Formular-Management-System *formdesk* zunächst eine Zuordnung der aufgetretenen Usability-Probleme zu den in 5.3 aufgeführten Problemkategorien sowie eine Identifizierung der Problemursachen.

Damit ergibt sich die in Tabelle 5.7 gezeigte Zuordnung von Problemkategorie und Ursache zu den jeweiligen aufgetretenen Problemen.

Mikrotask	Problem	Häufigkeit	Problemkategorie	Ursache
1	Proband findet Modellierungskonzepte schwer zu verstehen, benötigt Hilfe	5	KonzUnklar	8
2	kann den Workflow nicht ohne Hilfestellung modellieren	5	KonzUnklar	8
	kann den Workflow nicht ohne Hilfestellung aus der Aufgabenstellung extrahieren	5	KonzUnklar	8
6	weiß nicht, wie Layout zu übernehmen ist	4	KonzUnklar	2
	weiß nicht, wie ein Layout auszuwählen ist	2	KonzUnklar	2
	Es besteht Unklarheit darüber, wie das Layout auszuwählen ist und ob das angezeigte Formular ein Beispiellayout oder schon die Vorschau eines Formulars ist	3	OrientLosModProz	1
	Irritation über inkonsistente Bezeichnung Layout/Thema	1	UnsicherhRichtBedElem	9
7	probiert, im Bearbeitungsmodus den Wert durch ‘Enter’ drücken zu übernehmen	1	ProbBedLay	2
	findet das Konzept zur Bearbeitung des Titels nicht gut	2	KonzUnklar	6
	versucht, in Layoutvorschau Formularelemente anzuklicken	3	OrientLosModProz	1
	irritiert über Bezeichnung ‘Aufbewahren’	5	UnsicherhRichtBedElem	7
19	Findet im Bearbeitungsmodus nicht die Stelle, wo die Werte für die Radio-Buttons einzugeben sind	1	SuchBedElem	5
	irritiert über die Bedeutung der Option ‘alternative Wahl’	3	VerunsZahlrAusw	7
27	Proband findet nicht die Stelle zum Definieren der Abhängigkeit, Hinweis: Es ist eine Eigenschaft für das Element	2	SuchBedElem	2
	Hinweis notwendig: Es ist eine Voraussetzung für das Element, daraufhin orientiert sich der Proband im linken Menü	1	KonzUnklar	2
	versucht diese Bedingung bei den Radio-Buttons für die Priorität einzustellen	1	KonzUnklar	2
	Lösch-Icon bei den Bedingungen als solches nicht erkennbar	1	SuchBedElem	7
	Unklarheit über Bedeutung der Begriffe bei der Defintion der Bedingungen, z.B. ‘verriegeln’, ‘erlauben’, ‘sichtbar machen’, ‘unsichtbar machen’	4	UnsicherhRichtAktion	7
	(visuelles) Feedback über Funktion fehlt	2	UnsicherhRichtAktion	3
28	Unklarheit darüber, wo der Button im Formular erscheint	1	KonzUnklar	2
	Proband ist unsicher, wo der Button zu beschriften ist, probiert herum	5	OrientLosModProz	1
	sucht Button in Liste der Elemente, die hinzugefügt werden können, findet ihn dort nicht	4	OrientLosModProz	1
	Unklarheit darüber, ob Button vom System hinzu gefügt wird, oder ob man das selbst machen muss	1	OrientLosModProz	4
	findet in den Einstellungen nicht die Stelle, wo Beschriftung zu ändern ist	2	VerunsZahlrAusw	8

Mikrotask	Problem	Häufigkeit	Problemkategorie	Ursache
29	Proband erkennt nicht Unterschied Workflow-Bericht, Bericht an Benutzer des Formulars und E-Mail-Bericht	2	KonzUnklar	8
	Unklarheit darüber an wen der vom System erstellte vorhandene E-Mail Bericht geht	1	KonzUnklar	4
	sucht unter Einstellungen die Möglichkeit, Bestätigungs-E-Mail einzustellen	1	SuchBedElem	5
	Proband findet sich innerhalb des Menüs 'Email/Workflow' nicht zurecht	1	VerunsZahlrAusw	8
	Proband findet die Stelle, um Berichte einzustellen, nicht	2	SuchBedElem	1
	Irritiert über Bezeichnung E-Mail-Bericht	1	UnsicherhRichtBedElem	7
30	der Proband bemängelt, dass die linke Navigation innerhalb des Bearbeitungsmodus' von Berichten schlecht gemacht ist: ist nicht strukturiert, leitet den Proband nicht	1	ProbBedLay	5
	findet die Stelle wo Betreff des Berichts zu setzen ist nicht	3	SuchBedElem	5
	Setzt den Betreff unter 'Allgemein' -> Beschreibung	4	VerunsZahlrAusw	7
32	fügt Variable über Icon neben Betreff ein, daher erscheint Variable im Betreff und nicht im Inhalt	3	ProbBedLay	5
	braucht Hilfestellung zur Verwendung der Formularvariablen	2	KonzUnklar	2
	Proband hat große Schwierigkeiten mit dem Konzept der Formularvariablen	4	KonzUnklar	2
	Proband erkennt die Möglichkeit der Formularvariablen nicht	1	KonzUnklar	2
	Proband findet die Stelle zum Einfügen der Variablen nicht	3	SuchBedElem	5
33	Proband weiß nicht welche Art von Bericht erstellt werden soll	3	KonzUnklar	8
	Proband erkennt nicht die Möglichkeit des Versandkriteriums bei Workflow-Berichten	1	SU	8
	Proband findet nicht Stelle wo Berichte zu erstellen	1	OrientLosModProz	1
35	erstellt einen Bericht mit zwei Bedingungen für (gegensätzliche) Versandkriterien	1	KonzUnklar	2
	Proband sucht lange, bis er die Option des Versandkriteriums findet	1	SuchBedElem	5

Tabelle 5.7: *formdesk*: Aufgetretene Usability-Probleme

5.3.2.1 Anzahl der Usability-Probleme pro Mikrotask

Um eine Aussage über das Ausmaß der jeweils in den kritischen Mikrotasks aufgetreten Probleme treffen zu können, wird mithilfe von Tabelle 5.7 für jeden Mikrotask analysiert, zu wie vielen Usability-Problemen es in welcher Problemkategorie kam. Es wird dabei berücksichtigt, wie häufig ein Usability-Problem bei den Probanden auftrat.

Eine Zuordnung der aufgetretenen Usability-Probleme zu den einzelnen Problemkategorien ergibt die in Tabelle 5.8 gezeigte Verteilung.

Mikrotask	OrientLosModProz	SuchBedElem	UnsicherhRichtBedElem	UnsicherhRichtAktion	VerunsZahlrAusw	ProbBedLay	KonzUnklar	Summe
1	0	0	0	0	0	0	5	5
2	0	0	0	0	0	0	10	10
6	3	0	1	0	0	0	6	9
7	3	0	5	0	0	1	2	11
19	0	1	0	0	3	0	0	4
27	0	3	0	6	0	0	2	11
28	10	0	0	0	2	0	1	13
29	0	3	1	0	1	0	3	8
30	0	3	0	0	4	1	0	8
32	0	3	0	0	0	3	7	13
33	1	1	0	0	0	0	3	5
35	0	1	0	0	0	0	1	2
Summe	17	15	7	6	10	5	40	99

Tabelle 5.8: *formdesk*: Aufgetretene Usability-Probleme pro Mikrotask, gruppiert nach Problemkategorie

I In Tabelle 5.8 zeigt sich deutlich, dass es bei der Durchführung der beiden Mikrotasks 28 und 32 am häufigsten zu Problemen seitens der Probanden kam. Bei Mikrotask 28 sollte der Proband den Button zum Absenden des Formulars mit ‘Problem melden’ beschriften. Dazu war es notwendig, dass der Proband die Stelle im Modellierungsprozess findet, an der die Beschriftung des Buttons gesetzt werden kann. Die Probleme bei diesem Mikrotask sind nahezu alle in Problemkategorie ‘Orientierungslosigkeit im Modellierungsprozess’ einzuordnen, da die Probanden insofern nicht so vom System geführt wurden, dass ihnen klar war, dass der Button kein hinzuzufügendes Formular-Element ist, sondern dass dessen Beschriftung im Schritt ‘Einstellungen’ vorzunehmen ist. Die Probanden suchten also zunächst in den hinzuzufügenden Formular-Elementen, fanden dort aber kein Button-Element. Danach klickten sie sich suchend durch weitere Schritte des Modellierungsprozesses und fanden nur teilweise die entsprechende Stelle im Menü ‘Einstellungen’. Ebenso häufig traten Probleme bei Mikrotask 32 auf, bei dem von den Probanden gefordert war, die Bestätigungs-E-Mail an den Benutzer des Formulars zu personalisieren. An dieser Stelle musste der Proband die Möglichkeit der Formularvariablen erkennen, sowie deren Konzept verstehen und anwenden. Die meisten Probanden erkannten zwar die Möglichkeit der Formularvariablen, hatten jedoch aufgrund einer unübersichtlichen Gestaltung der Bearbeitungsoberfläche und schlecht erkennbarer Bedienelemente Probleme, das Konzept der Verwendung zu erfassen und zu benutzen.

II Bei der Durchführung von Mikrotask 27, bei dem es darum ging, dass der Proband eine Abhängigkeit von Formularelementen insofern modellieren sollte, dass ein bestimmter gesetzter Wert des einen Elements Voraussetzung dafür ist, dass ein zweites Element angezeigt wird. Zunächst gab es bei den Probanden das Problem, dass ihnen nicht intuitiv klar war, dass es die Möglichkeit gibt, Voraussetzungen für Elemente einzustellen. Diese Option war ein Menüpunkt im linken Menü innerhalb des Bearbeitungsmodus eines Formular-Elements. Sie wurde von den Probanden aufgrund der schlechten Strukturierung der Oberfläche innerhalb des Bearbeitungsmodus eines Formular-Elements jedoch nicht wahrgenommen. Weiterhin traten die Probleme hauptsächlich an

der Stelle auf, an der die Probanden die Bedingung für das Einblenden des abhängigen Feldes definieren mussten, da ihnen die vorhandenen Bezeichnungen der Kriterien nicht verständlich waren und fast immer erklärende Hinweise vom Testadministrator benötigten. Auch fühlen sich die Probanden dadurch verunsichert, dass die definierten Vorbedingungen eines Elements nicht in der Vorschau des Formulars wiedergespiegelt wurden.

III Mikrotask 7, bei dem die Probanden den Titel des Formulars bearbeiten sollten, war der erste durchzuführende Mikrotask bzgl. der Bearbeitung von Formularelementen nach der Auswahl des Layouts. Hier kam es fast immer zu dem Problem, dass die Probanden den Titel des Formulars in der Layoutvorschau bearbeiten wollten. Ihnen war vorher bei der Auswahl des Layouts nicht klar, dass das angezeigte Formular lediglich ein Beispiellayout und nicht die Vorschau eines Formulars ist. Dementsprechend erkannten sie auch nicht, dass sie vorher den Schritt der Layout-Auswahl nicht vollständig abgeschlossen hatten. Dies ist darauf zurück zu führen, dass sie vom System nicht durch die einzelnen Schritte des Modellierungsprozesses geführt wurden. Sie konnten nicht erkennen, dass es diese Schritte der Layoutauswahl, der Bearbeitung von Formularelementen, dem Vornehmen von weiteren Formular-Einstellungen sowie dem Erstellen verschiedener Berichte überhaupt gibt. Der Testadministrator musste bei fast allen Probanden einen Hinweise geben, damit sie vom Schritt der Layoutauswahl zum Schritt des Hinzufügens und Bearbeitens von Formularelementen kamen. Bei der Bearbeitung des Titels gab es dann starke Verunsicherung der Probanden beim Übernehmen des bearbeiteten Titels, da den Probanden die Bezeichnung ‘Aufbewahren’ zum Speichern ihrer vorgenommenen Änderung nicht geläufig war, sie jedoch auch keine andere Option sahen, ihre Eingaben zu übernehmen.

IV Neben den eben beschriebenen Problemen bei Mikrotask 7, die die Führung des Probanden durch die verschiedenen Schritte des Modellierungsprozesses betreffen, gab es schon innerhalb des ersten Modellierungsschrittes, der Auswahl eines Layouts, in Mikrotask 6 das Problem, dass die Probanden innerhalb dieses Schrittes nicht geführt wurden und durch eine schlecht strukturierte und sehr komplex gestaltete Oberfläche überfordert waren und das Konzept zur Auswahl eines Layouts nicht erfassen konnten.

V Das Extrahieren und Modellieren des Formular-Workflows war Bestandteil von Mikrotask 2. Hier war eine erhebliche Transferleistung des Probanden von der Aufgabenstellung zum Workflow des Formulars gefordert. Keiner der Probanden konnte dieses ohne Hilfestellungen des Testadministrators umsetzen. Die Modellierung des Workflows war für die Probanden in keiner Weise intuitiv und erschien ihnen als zu komplex und nicht notwendig für die einfachen Prozesse des Formularwesens.

VI Bei Mikrotask 29 war gefordert, dass der Proband eine Bestätigungs-E-Mail aktiviert, welche an den Benutzer des Formulars nach Abschicken des Formulars gesendet wird. Zunächst hatten die Probanden auch hier wieder Probleme, den Schritt im Modellierungsprozess zu erkennen, was auch an dieser Stelle wieder auf die extrem überlade-

ne und schlecht strukturierte Oberfläche im Modellierungsschritt ‘Einstellungen’ zurück zu führen war. Zudem war die Bezeichnung ‘Berichte’ für die Probanden irreführend. Außerdem verstanden die Probanden das Konzept der verschiedenen Arten von Berichten nicht. Es mussten oft Erklärungen zu den unterschiedlichen Berichtarten (E-Mail-Bericht, Bericht an Benutzer des Formulars und Workflow-Bericht) vom Testadministrator geliefert werden.

VII Hatte der Proband den Bericht an den Benutzer des Formulars aktiviert, so war es in Mikrotask 30 gefordert, den Betreff für diesen Bericht zu setzen. Dabei bestand auch hier wieder das Problem, dass die Probanden den Betreff zunächst an der falschen Stelle, nämlich unter den allgemeinen Eigenschaften in der Beschreibung des Berichts einfügten. Zurückzuführen war auch dieses auf die zu komplex gestaltete und schlecht strukturierte Oberfläche des Bearbeitungsmodus der Berichte. Oft war der Hinweis an die Probanden notwendig, dass sie sich innerhalb des Bearbeitungsmodus des Berichts in dem linken Menü orientieren müssen um dort die entsprechende Stelle für die Bearbeitung der Berichtinhalte zu finden.

5.3.2.2 Gruppierung der Usability-Probleme nach Problemkategorie

Um für das Produkt *formdesk* zu erkennen, in welchen Bereichen es hauptsächlich zu Problemen in der Usability kommt, wurde anhand der in Tabelle 5.7 aufgeführten Daten für jeden kritischen Mikrotask die Anzahl der Usability-Probleme pro Problemkategorie summiert, was wie auch in Abschnitt 5.3.1.2 der letzten Zeile in Tabelle 5.7 entspricht.

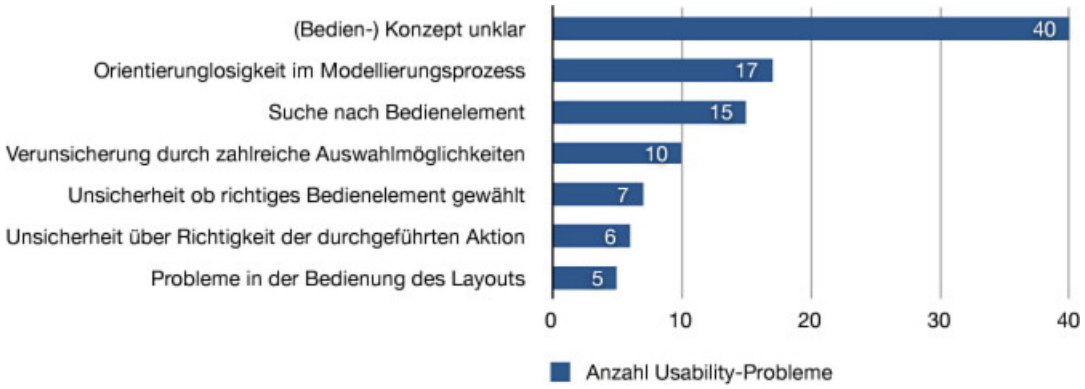


Abbildung 5.5: *formdesk*: Summe der aufgetretenen Usability-Probleme pro Problemkategorie

In Abbildung 5.5 wird deutlich, dass der weitaus größte Teil der Probleme dadurch auftrat, dass die Probanden mit dem Konzept bestimmter Funktionen nicht zurecht kamen. Zunächst ist dabei das in Mikrotask 1 und 2 geforderte Konzept der BPMN und der Modellierung des Formular-Workflows zu nennen. Die Probanden benötigten alle eine vom Testadministrator

unterstützte und erklärende Einführung in die Konzepte der BPMN, wobei die Probanden die Modellierungskonstrukte als sehr abstrakt und wenig intuitiv empfanden. Wie bereits unter I erläutert, stieß die Idee dieser Art der Workflowmodellierung auf Abneigung bei den Probanden und erweckte bei allen das Gefühl der Unsicherheit und Unbehaglichkeit. Wie bereits unter IV erwähnt, gab es bei der Auswahl des Formularlayouts bei den Probanden weitestgehend Probleme mit dem Bedienungskonzept.

Ebenfalls große Probleme mit dem Verständnis des Konzepts hatten die Probanden bei der unter III angesprochenen Aufgabe, ein bestimmtes Formularelement nur in Abhängigkeit von dem Wert eines anderen Formularelements einzublenden. Es fiel ihnen schwer zu erkennen, an welcher Stelle diese Abhängigkeit eingestellt werden muss, bzw. dass dieses eine Vorbedingung für das einzublendende Formularelement ist. Wie unter VI erwähnt, gab es bei den Probanden Unklarheit und Verwirrung über die verschiedenen Berichtsarten, wie diese zu verwenden sind und wann welche Art von Bericht für welche E-Mail-Benachrichtigung erstellt werden muss. Die Probanden waren mit dem Konzept der unterschiedlichen Berichte und deren Verwendung überfordert. Zur Personalisierung der E-Mail-Bestätigung an den Benutzer des Formulars, musste das Konzept der Formularvariablen erkannt und verstanden werden. Dies war jedoch wie unter I aufgezeigt größtenteils nicht ohne Hilfestellung des Testadministrators möglich.

Probleme, die die Orientierungslosigkeit der Probanden im Modellierungsprozess betrafen, traten hauptsächlich dadurch auf, dass die Probanden nicht durch den Prozess der Formularerstellung und der weiteren vorzunehmenden Schritte geführt wurden. Wie im vorherigen Abschnitt erläutert, war den Probanden nicht klar, welche Schritte und Optionen die Auswahl eines Layouts umfasst, welche Einstellungen dabei vorzunehmen sind und wie der Schritt der Layoutauswahl abzuschließen ist. Keiner der Probanden erkannte, dass die oberen Registerkarten der Oberfläche die verschiedenen zu durchlaufenden Schritte bei der Formular- und Workflow-Erstellung repräsentieren. Auch bestand bei den Probanden Unklarheit darüber, ob und an welcher Stelle der Submit-Button des Formulars vom System eingefügt wird oder ob dieser als ein Formularelement von ihm selbst hinzugefügt und positioniert werden muss.

In der Kategorie ‘Suche nach Bedienelement’ wurden die Probleme durch die ausführlich im vorhergehenden Abschnitt besprochenen Mikrotasks 28, 29, 30 und 32 beigesteuert. Diese traten dadurch auf, dass die Gestaltung der Oberflächen innerhalb des Workflowschrittes ‘Einstellungen’, innerhalb der Bearbeitung von E-Mail-Berichten und innerhalb des Bearbeitungsmodus von Formularelementen vom Probanden nicht als Führung durch die verschiedenen Konfigurationsschritte und -optionen wahrgenommen wurde. Innerhalb der einzelnen Konfigurationsschritte waren die Probanden durch die vielen Optionen überfordert, wussten diese nicht zu deuten und fühlen sich durch diese Komplexität stark verunsichert.

Die Probleme aus der Kategorie ‘Unsicherheit ob richtiges Bedienelement gewählt’ sind auf das schlechte Wording des Produkts sowie auf ein teilweise etwas inkonsistentes Bedienungskonzept zurück zu führen. Die Verunsicherungen der Probanden über die Richtigkeit

ihrer durchgeführten Aktion traten hauptsächlich bei der Formulierung der Versandkriterien der Workflow-Berichte und den Voraussetzungen für Formularelemente auf, da dort den Probanden die Bedeutung der verschiedenen Begriffe nicht klar war. Die sehr komplexe Gestaltung der Berichte und hinzu kommend die teilweise schlechte Bezeichnung einzelner Optionen führte zu starker Verunsicherung der Probanden. Alle Probanden bemängelten die schlechte Bedienbarkeit insofern, dass die Oberfläche des Produkts unstrukturiert ist und einen nicht leitet. Diese unübersichtliche Gestaltung der Oberfläche hatte auch zur Folge, dass teilweise Bedienelemente nicht wahrgenommen wurden.

5.3.2.3 Auswertung der Interviews

Die Auswertung der Interviews welche im Anschluss an die Testdurchführungen mit der Anwendung *formdesk* durchgeführt wurden, lieferten Erkennisse darüber, wie die Probanden das Arbeiten mit der Anwendung *formdesk* empfanden.

Die Modellierungskonzepte der BPMN wurden von den Probanden durchweg als zu schwer, zu abstrakt und nicht intuitiv bezeichnet. Auch waren sie der Meinung, dass für die überschaubaren Workflows des Formularwesens eine Modellierung nicht angemessen und notwendig ist, zudem ihnen der Workflow in der Form auch nicht in der Anwendung präsent war.

Durchweg positiv bewerteten die Probanden die umfangreiche Funktionalität des Produktes (Voraussetzungen, Bedingungen, Gestaltung von E-Mail-Inhalten, Versandkriterien), welche weitreichende Anforderungen in ihrer Problemdomäne abdeckt. Dem gegenüber steht jedoch neben dem schlechten Design und dem insgesamt sehr negativen Eindruck des Systems die schlecht strukturierte und extrem verwirrend gestaltete Oberfläche. Die Benutzung des Produkts machte den Probanden keinen Spaß, da sie sich schwer verständlich und nicht intuitiv gestaltete. Das Vorgehenskonzept der meisten Funktionen wurde als zu kompliziert und komplex wahrgenommen, was dazu führte, dass die Probanden lange suchen mussten bevor sie entsprechend ihre Aufgabe umsetzen konnten.

Die Oberfläche beschrieben alle Probanden als sehr unübersichtlich, Funktionen sind schlecht strukturiert und gruppiert. Es gibt generell zu viele Bedienelemente auf der Oberfläche (oben, unten, links, rechts). Hinzu kommen schlecht gestaltete Icons, deren Bedeutung nicht klar wird, ein extrem schlechtes Wording des Produkts und wenig hilfreiche Hinweistexte.

Das Produkt unterstützte den Probanden durch Fehlermeldungen bei Falscheingaben. Die Probanden begrüßten den ‘Schritt für Schritt’- Ansatz des Produkts, fühlten sich jedoch nicht gut durch diesen geleitet. Ihnen fehlte die Führung durch die Arbeitsschritte und sie hatten keine Orientierung, in welchem Bearbeitungsschritt sie sich gerade befinden.

Des Weiteren wurden die prinzipiell vorhandenen erweiterten Layoutmöglichkeiten des Formulars sowie dessen Vorschau und Übersicht am Ende des Modellierungsprozesses als sehr

positiv bewertet. Manche Probanden waren der Meinung, dass, nachdem das Bedienungskonzept verstanden wurde, wohl eine recht gute, aber aus Aspekten der Bedienung auch sehr kompromissbehaftete Arbeit mit dem Produkt möglich wäre.

Als Verbesserungspotential zu dem Produkt sahen die Probanden eine übersichtlichere Gestaltung sowie eine bessere Beschriftung der Menüpunkte und Icons. Sie empfänden eine Übersicht, welche dem Benutzer verdeutliche, an welcher Stelle des Modelleriungsprozesses er sich gerade befindet, sehr hilfreich. Generell fänden sie es besser, wenn die Funktionen übersichtlich innerhalb eines Oberflächendialogs angeordnet wären und sie nicht im Bearbeitungsmodus von Elementen oder Berichten beispielsweise auf eine neue Oberfläche geleitet werden, welche sich alleinig auf das bearbeitete Element bezieht und damit der Kontext des Formulars verlassen wird. Angenehmer empfänden es die Probanden, wenn sie jederzeit eine integrierte Sichtweise des Elementes im Formular hätten. Auch sehen die Probanden entsprechendes Verbesserungspotential in der Führung durch den Modellierungsprozess, beispielsweise in Form eines gut leitenden Wizards.

5.3.3 *WebDSL*

Wie in Abschnitt 5 bereits erläutert, lieferte die Testdurchführung mit der domänenspezifischen Sprache *WebDSL* keine Testdaten in Form von zu analysierenden Mikrotasks. Die erhobenen Daten bestanden aus den Video- und Audioaufzeichnungen der Äußerungen der Probanden, während sie versuchten, die Aufgabenstellung umzusetzen.

Zunächst erkannten alle Probanden die in den Beispielen vorhandenen Strukturen wie page, table, row, action und form. Dies ist darauf zurück zu führen, dass diese Begriffe und ihre Funktion eindeutig bezeichnet sind und insbesondere Benutzer mit HTML-Kenntnissen diese leicht wieder erkennen.

Jedoch waren sich auch alle Probanden dahingehend einig, dass sie zwar die groben Strukturen erkennen können, sie jedoch kein Verständnis der näheren Funktionen und Details haben. Dazu wäre ihrer Meinung nach eine intensive Beschäftigung und ein Erlernen der Zielsprache notwendig. Dazu wäre jedoch nur in einem Fall die prinzipielle Bereitschaft vorhanden gewesen. Die weiteren Probanden sahen die Dokumentation als zu komplex, empfanden Abneigung und Überforderung und sahen diese nicht als geeignet und informationsliefernd an. Sie orientierten sich eher an den Beispielen, wobei alle der Meinung waren, dass es Ihnen nicht möglich wäre, eigene Aufgaben umzusetzen, sondern dass sie maximal ein Beispiel kopieren und versuchen würden, dieses anzupassen. Dazu wäre jedoch visuelles Feedback notwendig.

Die Probanden sahen bei dieser Form der Modellierung alle die Vorteile der vielfältigen Möglichkeiten und Freiheiten in der Umsetzung sowie dem ggf. gegebenen Geschwindigkeitsvorteil bei der Umsetzung auf Code-Ebene. Sie empfanden jedoch größtenteils große bis absolute Abneigung gegenüber diesem Prinzip. Sie waren der Meinung, dass dies eine intensive Beschäftigung mit der Sprache erfordern würde, die sie aber für ihren Problembereich nicht für angemessen halten. Für einen Teil der Probanden stellte die textuelle Modellierung keine Alternative für ihren Arbeitsbereich dar. Andere Probanden konnten sich eine derartige Modellierung vorstellen, wenn ihnen gleichzeitig eine entsprechende visuelle Oberfläche geboten wird, die sie dabei unterstützt, führt und entsprechendes visuelles Feedback bietet.

5.4 Fazit

Aus den vorangegangenen Abschnitten der Auswertung und Analyse der Testdurchführungen lassen sich die folgenden Stärken und Schwächen der einzelnen Produkte erkennen:

5.4.1 *jotform*

Stärken:

jp1 Idee des Drag-and-Drop Bedienungskonzeptes

jp2 Alle Funktionen und Elemente des Formulars werden jederzeit übersichtlich dargestellt

jp3 Die Anzeige und Bearbeitung der Elemente im Eigenschaften-Inspektor ist übersichtlich und intuitiv

jp4 Das Produkt setzt auf gängige Bedienkonzepte

jp5 Funktionen sind gut auffindbar

jp6 Einfache Bedienung

jp7 Optisch ansprechend

jp8 Gut strukturierte Oberfläche

Schwächen:

jn1 Das Bedienkonzept ist schlecht umgesetzt.

jn2 Das Konzept der Anzeige der unterschiedlichen Eigenschaften (Formular und Elemente) ist nicht erkennbar für den Benutzer.

jn3 Grundlegende Funktionen fehlen.

jn4 Keine Fehlermeldung bei falsch gesetzten Optionen bzw. syntaktisch falschen Eingaben.

jn5 Teilweise fehlendes Feedback zu umgesetzten Funktionen.

jn6 Fehlende Informationen zu implizit vom System aktivierten Optionen.

jn7 Teilweise inkonsistentes Bedienungskonzept.

jn8 Optionen sind teilweise sehr versteckt.

jn9 Der Wizard führt den Benutzer nicht und vermittelt ihm nicht, an welcher Stelle des Modellierungsprozesses er sich gerade befindet.

jn10 Teilweise sehr technische oder wenig aussagekräftige Bezeichnungen.

jn11 Keine erläuternden Hinweise, z.B. bei Mouse-Over.

5.4.2 *formdesk*

Stärken

fp1 umfangreiche Funktionen

fp2 'Schritt für Schritt' - Ansatz des Modellierungsprozesses

fp3 Unterstützung verschiedenster E-Mail-Benachrichtigungen sowie deren Personalisierung

fp4 Dynamisierung des Formulars durch Anzeigekriterien für Formularelemente möglich

fp5 Fehlermeldungen bei Falscheingaben

fp6 Gute Gestaltungsmöglichkeiten für Layout

Schwächen:

fn1 Extrahieren des Workflows aus der Aufgabenstellung zu schwer, zu abstrakt

fn2 Modellierung des Workflows in BPMN zu komplex, nicht angemessen

fn3 Keine Führung des Benutzers durch den Modellierungsprozess

fn4 Einzelne Schritte des Prozesses nicht als solche erkennbar, fehlende Orientierung während der Modellierung

fn5 Zur Bearbeitung von Elementen und Berichten wird der Kontext des Formulars verlassen

fn6 Konzept der Bearbeitung von Elementen und Berichten zu komplex

fn7 Navigation und Konzept des Bearbeitungsmodus von Elementen und Berichten ist nicht nachvollziehbar

fn8 Konzept der verschiedenen Berichtarten zu komplex

fn9 Konzept der Formularvariablen zu komplex und schlecht vermittelt

fn10 Funktionen von Icons sehr schwer erkennbar

fn11 Funktionen schlecht strukturiert und gruppiert

fn12 Oberfläche überladen, zu komplex, zu viele Bedienelemente, Icons sehr schwer auffindbar, nicht intuitiv

fn13 Fehlendes Feedback zu bestimmten Optionen

fn14 Schlechte Bezeichnungen

fn15 Teilweise inkonsistentes Bedienungskonzept

5.4.3 *webdsl*

Stärken:

wp1 Strukturen gut erkennbar

wp2 Aussagekräftige Begriffe

wp3 Flexibilität bzgl. der Möglichkeiten

wp4 Geschwindigkeitsvorteil

Schwächen:

wn1 Zielsprache muss erst erlernt werden

wn2 Modellierungskonzept der textuellen Notation stellt keine Alternative dar

wn3 Teilweise Modellierungskonzept im Zusammenhang mit visuellem Feedback akzeptabel

wn4 Zu komplex in der Anwendung

wn5 Probanden könnten keine Anwendung/keinen Workflow selbst modellieren

wn6 Für Formularwesen als nicht angemessen empfunden

5.4.4 Schlussfolgerungen

Anhand der erkannten Probleme und entprechenden Stärken und Schwächen lassen sich die folgenden Aussagen treffen:

1 Die Modellierung in textueller Notation ist für die meisten Domänenexperten nicht akzeptabel und stößt auf Ablehnung. (wn2, wn5 und wn6)

2 Für die Prozesse des Formularwesens empfinden die Domänenexperten eine Modellierung des Workflows als zu schwer und nicht angemessen. (fn1, fn2)

3 Die Domänenexperten möchten ihr Formular anhand aussagekräftiger und gut bezeichneter Icons und Menüpunkte ‘zusammen klicken’ und ihre funktionalen Anforderungen direkt im Formular bzw. an den entsprechende Elementen im Formular abbilden. (jp1, jp6, jn10, fn14, fp1)

4 Einfache Formulargeneratoren sind teilweise gut und intuitiv bedienbar, haben aber gravierende Mängel bzgl. der von den Domänenexperten benötigten Funktionalität.(jp2, jp3, jp4, jp5, jp7, jp8, jn3, fp3, fp4)

5 Produkte zur Workflowmodellierung sind in ihrem Funktionsumfang für die Domäne der einfachen Prozesse in Webportalen zu komplex und überfordern den Domänenexperten durch ihre komplizierten Konzepte und durch übermässigen Funktionsumfang. (fn7, fn8, fn9, fn12)

6 Durch ihren hohen Funktionsumfang zeigt sich bei Produkten für die Workflowmodellierung der strukturelle Effekt, dass sie aufgrund der übermäßigen Komplexität in ihrer Repräsentation teilweise die für den Domänenexperten relevanten Informationen aufgrund anderer Informationen vernachlässigen bzw. nicht optimal darstellen können. Diese wurde unter [Gre96], im Abschnitt 2.2.1 Psychologie des Programmierens - Mentale Repräsentation, als Hauptproblem für das Erlernen von Programmen erkannt. Auch kommt es dadurch zu der ebenfalls im Abschnitt 2.2.1 unter den ‘Maximen bzgl. der Untersuchung von strukturellen Effekten in der mentalen Verarbeitung’ beschriebenen Dislokation und Zerstreuung der für den Domänenexperten relevanten Funktionen. (fn5, fn6, fn10, fn11)

7 Für die Domänenexperten ist es wichtig, von der Anwendung durch die einzelnen Schritte der Formular-Erstellung und Prozessmodellierung geführt zu werden und jederzeit eine Orientierung zu haben, in welchem Schritt sie sich gerade befinden (siehe Psychologie des Programmierens, Effekt der Umgebung). (fp2, fn3, fn4, jn9)

8 Für die Domänenexperten ist ein inkonsistentes Bedienungskonzept sehr problematisch (siehe Framework der kognitiven Dimensionen im Abschnitt 2.2.3 Punkt ‘Konsistenz’). (jn7, fn14)

9 Die Anwendung soll den Domänenexperten durch entsprechende Vorgaben möglichst vor Falscheingaben bewahren oder wenn dieses nicht möglich ist, ihn durch erklärende Fehlermeldungen oder anderweitiges visuelles Feedback darauf aufmerksam machen (siehe Framework der kognitiven Dimensionen, Fehleranfälligkeit). (jn4, jn5, fp5, fn13)

Aus den oben dargestellten Herleitungen ist erkennbar, das in allen Produkten die Anforderungen der Domänenexperten an die Bedienung und Funktionalität in unterschiedlicher Ausprägung nicht ausreichend umgesetzt sind. Weder der Formulargenerator noch ein komplexes Formular-Management-System haben an dieser Stelle die optimale Methode zur Prozessmodellierung in Webportalen dargestellt.

6 Produktvision

Die im Rahmen der Evaluierung betrachteten Alternativen unterschiedlicher Produkte und Produktansätze und deren Analyse haben ergeben, dass weder die Produkte zur einfachen Erstellung von Formularen, noch die vorhanden Produkte für das erweiterte Formularmanagement den Anspruch einer einfachen und intuitiven Modellierung der Prozesse in Webportalen erfüllen.

Wie im letzten Abschnitt herausgestellt, bieten die einfachen und auch recht intuitiven HTML-Formulargeneratoren nicht, oder nur sehr rudimentär, die Möglichkeit der Modellierung von Formularabläufen. Es wurde auch kein Produkt gefunden, welches hinreichend gut und einfach die Möglichkeit bietet, die beschriebenen Formularabläufe zu modellieren. Insbesondere Produkte, welche auf dem Hosting von Formularen basieren, sind problematisch hinsichtlich der Integration der Workflows und Formulare in bestehende Webportale. Anforderungen, welche auf der Anbindung eines Verzeichnisdienstes oder anderen portalinternen Eigenschaften basieren können, werden von diesen Produkten nicht umgesetzt.

Ein Produkt, welches eine einfache und intuitive Methode zur Modellierung der Prozesse in Webportalen vor dem Hintergrund der im Rahmen dieser Arbeit definierten Anforderungen bereitstellen soll, muss die Integration portalinterner Funktionen unterstützen. Die Anwendung muss in das Portal integrierbar sein und mit der Umgebung und IT-Infrastruktur des Portalbetreibers kommunizieren können. Es soll eine 'Back-End'-Repräsentation des Formulars und der eingegangenen Daten abrufbar sein, welche auch bestimmte Funktionalitäten auf diesen Daten anbietet. Dafür müssen Objekte, die Gegenstand der Formulare sind, beispielsweise Veranstaltungen oder Veranstaltungstermine und eingegangene Formulardaten in einer portalinternen Datenbank abgebildet werden.

Die Modellierung von Prozessen außerhalb der Domäne des Formularwesens in Webportalen steht ausdrücklich nicht im Fokus der zu entwickelnden Methode. Das Produkt soll sich durch eine geringe Komplexität auszeichnen und keinerlei Modellierung unternehmensinterner oder -übergreifender Geschäftsprozesse unterstützen.

Die zu entwickelnde Anwendung wird konzipiert sein für die Benutzung durch Domänenexperten, welche mit dem Konzept von HTML-Formularen, deren Elementen sowie mit Formularabläufen vertraut sind, jedoch nicht über die programmiertechnischen Kenntnisse verfügen, diese selbst zu realisieren oder in ein Webportal zu integrieren.

Im Folgenden bezeichnet der Begriff '**Benutzer**' den Domänenexperten, der mit der Methode bzw. der Anwendung zur Modellierung der Geschäftsprozesse in Webportalen arbeitet. Dem gegenüber stehen die Begriffe '**Formular-Benutzer**' oder '**Benutzer des Formulars**', welche den Anwender bezeichnen, der das von dem Domänenexperten erstellte Online-Formular verwendet.

6.1 Funktionale Anforderungen

Die funktionalen Anforderungen werden in Form von in [Coh04] beschriebenen User Stories erhoben. Da im Rahmen dieser Arbeit eine vollständige Umsetzung aller Funktionen nicht möglich ist, sind die Anforderungen in zwei Kategorien unterteilt. In die erste Kategorie fallen funktionale Anforderungen, die im Prototypen zumindest exemplarisch umzusetzen sind, da diese die technische Realisierbarkeit des entwickelten Konzepts aufzeigen.

In die zweite Kategorie fallen die Anforderungen, die zwar konzeptionell im Prototypen berücksichtigt werden müssen, deren Implementierung jedoch zum Aufzeigen der Funktionsweise des Konzepts nicht relevant sind.

Innerhalb der Kategorien entspricht die Aufzählungsreihenfolge der funktionalen Anforderungen einer vorgenommenen Priorisierung der Anforderungen.

6.1.1 Exemplarisch umzusetzende funktionale Anforderungen

1 Als Benutzer kann ich ein neues HTML-Formular erstellen.

2 Als Benutzer kann ich Formularelemente zu einem Formular hinzufügen.

3 Als Benutzer kann ich die Eigenschaften eines Formularelements bearbeiten.

4 Als Benutzer kann ich eine Validierung für E-Mail-Adressen bei Eingabefeldern festlegen.

5 Als Benutzer kann ich Kriterien für die dynamische Anzeige von Formularelementen angeben.

6 Als Benutzer kann ich E-Mail-Bestätigungen an die Benutzer des Formulars aktivieren.

7 Als Benutzer kann ich für das Verschicken der E-Mail-Bestätigungen Kriterien basierend auf Formularwerten definieren.

8 Als Benutzer kann ich E-Mail-Benachrichtigungen für den Empfang der Formulardaten aktivieren.

Hier liegt zunächst die höchste Priorität auf den beiden Funktionen, welche dem Benutzer ermöglichen, ein Formular zu erstellen. Danach ist es wichtig für das Konzept, dass der Formular-Benutzer die Formularelemente bearbeiten kann. Dazu zählt auch die Angabe einer Validierung für den Wert eines Eingabefeldes, exemplarisch soll hier die Validierung einer E-Mail-Adresse sein. Auch die Formulierung von Bedingungen für die Anzeige von Formularelementen ist ein wichtiger Aspekt der entwickelten Methode, welcher im Prototypen repräsentiert sein muss. Ebenso müssen die neben der Erstellung des Formulars wichtigen Konzepte der E-Mail-Benachrichtigungen und -Bestätigungen exemplarisch umgesetzt werden, um deren Realisierbarkeit entsprechend der entwickelten Methode zu untermauern.

6.1.2 Im Prototypen konzeptionell vorzusehende funktionale Anforderungen

1 Als Benutzer kann ich Formularelemente innerhalb des Formulars frei positionieren.

2 Als Benutzer kann ich Formularelemente von einem Formular entfernen.

3 Als Benutzer werden mir ungültige Eingaben bei der Bearbeitung von Formularelementen gemeldet.

4 Als Benutzer bekomme ich vom System mögliche Werte für Formularelemente zur Auswahl dargestellt.

5 Als Benutzer kann ich ein Formular speichern.

6 Als Benutzer kann ich ein gespeichertes Formular öffnen und bearbeiten.

7 Als Benutzer kann ich ein Formular in mein Webportal integrieren.

8 Als Benutzer kann ich ein Formular aus dem Portal löschen.

9 Als Benutzer kann ich für das Verschicken der E-Mail-Benachrichtigungen Kriterien basierend auf Formularwerten definieren.

10 Als Benutzer kann ich E-Mails zu einem definierten Zeitpunkt verschicken lassen.

11 Als Benutzer kann ich für eingehende Formulardaten Dispatching-Regeln angeben.

12 Als Benutzer kann ich für ein Formular Stellvertreterregelungen angeben.

13 Als Benutzer kann ich interne Benutzerdaten automatisch vervollständigen lassen.

14 Als Benutzer kann ich die E-Mail-Bestätigungen an die Benutzer des Formulars mit Formularwerten dynamisieren.

15 Als Benutzer kann ich mir in einer Übersicht die eingegangenen Formulardaten anzeigen lassen.

16 Als Benutzer kann ich eine Validierung der Werte von Eingabefeldern festlegen.

17 Als Benutzer kann ich eine Vorbelegung für bestimmte Formularelemente angeben.

18 Als Benutzer kann ich eine Übersicht der eingegangenen Formulardaten per E-Mail verschicken.

19 Als Benutzer kann ich kann ich einen Workflow für eingegangene Formulardaten auswählen.

20 Als interner Formular-Benutzer kann ich mir die Daten meines abgesendeten Formulars anzeigen lassen.

21 Als interner Formular-Benutzer kann ich die Daten meines abgesendeten Formulars bearbeiten.

Auf höchster Prioritätsstufe stehen hier zunächst die Funktionen, um die Bearbeitungsmöglichkeiten von Formularelementen zu vervollständigen, da die Bearbeitung von Formularelementen die von den Benutzern am häufigsten durchgeführte Aktion sein wird. Danach folgen die Funktionen, welche die Persistierung und Bearbeitung von Formularen, sowie die Integration in das Portal betreffen, da diese für die Arbeit des Benutzers mit dem System essentiell sind. Als nächstes kommen die User Stories, welche nicht mehr die Formularbearbeitung an sich, sondern die Vervollständigung der Funktionen bzgl. der E-Mail-Benachrichtigungen betreffen. An letzter Stelle stehen die User Stories, welche die Funktionalität des Back-Ends, also der Verarbeitung der eingegangenen Formulardaten, betreffen.

6.2 Deklarative Beschreibung der Produkteigenschaften

Neben den im ISO/IEC 9126 Modell für Softwarequalität aufgeführten Qualitätsmerkmalen [swq] haben sich aus der Evaluierung und Analyse bestehender Produkte weitere nichtfunktionale Anforderungen für die zu entwickelnde Methode ergeben. Zum einen soll der Formular-Workflow nicht als solcher vom Benutzer modelliert werden müssen. Auch wurde eine Modellierung der Formulare und des Workflows in textueller Notation von den Domänenexperten als unangemessen angesehen. Die Domänenexperten bevorzugen eine WYSIWYG-Darstellung des Formulars, in der sie direkt ihre Anforderungen an den Formularelementen bzw. dem Formular bearbeiten können. Es wurde durch die Usability-Tests noch einmal deutlich, wie wichtig dabei eine aussagekräftige und nicht zu technische Bezeichnung der Icons und Bedienelemente ist. Ebenso hat sich gezeigt, dass das Produkt ausschließlich die für die Domänenexperten relevanten Funktionen bereitstellen sollte, um eine Zerstreuung und Dislokation dieser Funktionen zu vermeiden. Denn wie sich mehrfach in den Testdurchführungen zeigte, führt dies zu starker Verunsicherung der Benutzer und entsprechend langer Suche nach Bedienelementen.

Als ein sehr wichtiger Aspekt zeigte sich, dass die Anwendung den Benutzer schrittweise durch den Modellierungsprozess führt. Dabei muss dem Benutzer in jedem Schritt klar sein, an welcher Stelle des Modellierungsprozesses er sich gerade befindet, welche Schritte noch kommen und was diese Schritte an Optionen beinhalten. Es hat sich auch herausgestellt, dass die Benutzer es als vorteilhaft empfinden, wenn zur Bearbeitung der Formularelemente nicht der Fokus des gesamten Formulars verlassen wird, sondern wenn sie auch während der Bearbeitung der Eigenschaften der Formularelemente jederzeit einen Überblick über das gesamte Formular haben und dort sofortiges visuelles Feedback zu ihren vorgenommenen Einstellungen bekommen. Das Drag-and-Drop Bedienungskonzept wurde von allen Probanden der Usability-Tests als sehr angenehm und intuitiv empfunden, was entsprechend in der Entwicklung berücksichtigt werden sollte. Es sollten nicht zu viele unterschiedliche Bedienelemente (links, rechts, oben, unten) auf der Oberfläche platziert sein, da diese von den Benutzern nicht, oder nur sehr schlecht wahrgenommen werden.

7 Das konzeptuelle Modell des Benutzerinterfaces

In den vorangegangenen Abschnitten wurden bestehende Produkte für die Domäne der Prozessmodellierung in Webportalen evaluiert und in ihren Stärken und Schwächen analysiert. Die Ergebnisse dieser Analyse bildeten daraufhin die Grundlage für die Formulierung einer Produktvision, welche eine verbesserte Methode zur intuitiven Prozessmodellierung in Webportalen darstellen soll. Für diese Methode gilt es nun, unter Berücksichtigung der erkannten Usability-Aspekte, das Konzept des Benutzerinterfaces zu entwerfen.

Die zu entwickelnde Anwendung wird thematisch gesplittet sein, der zentrale Teil betrifft die leichte und intuitive Modellierung der Formulare und deren Prozesse, während der andere Teil die Verwaltung und Anzeige vorhandener Formulare und eingegangener Formulardaten beinhaltet. Da der zweite Teil jedoch nicht im Fokus dieser Arbeit steht, wird im Folgenden das Konzept und der Entwurf des ersten Teils betrachtet.

Durch die Evaluierung und Analyse bestehender Systeme haben sich im Rahmen der Durchführung und der Analyse der Usability-Tests entscheidende Facetten einer zu entwickelnden Methode für die Domäne der Prozesse in Webportalen herauskristallisiert. Demnach sind beim Entwurf des Interface Konzepts folgende Punkte zu beachten:

1 Die Benutzer bevorzugen eine WYSIWYG-Darstellung des Formulars.

2 Die Benutzer möchten ihre Funktionalität direkt an den Formularelementen selbst bearbeiten.

3 Die Benutzer möchten eine schrittweise Führung durch den Modellierungsprozess: welche Schritte gibt es und welcher Schritt beinhaltet welche Optionen.

4 Die Benutzer benötigen jederzeit eine Orientierung im Modellierungsprozess: in welchem Schritt befinden sie sich gerade, welche Schritte sind bereits durchlaufen und welche Schritte folgen noch.

5 Die Benutzer möchten zwischen den einzelnen Schritten navigieren können.

6 Die Benutzer möchten zur Bearbeitung der Formularelemente nicht den Fokus des Formulars verlassen.

7 Die Benutzer möchten jederzeit einen Überblick über das gesamte Formular haben.

8 Die Benutzer möchten jederzeit alle Funktionen des Bearbeitungsschrittes, in dem sie sich gerade befinden, im Überblick haben.

9 Die Benutzer möchten nicht zu viele unterschiedliche Bedienelemente (Menü oben, unten, links, rechts) haben.

10 Die Benutzer empfinden das Drag-and-Drop Bedienungskonzept als sehr angenehm.

Abbildung 7.1 liefert einen Überblick über das Vorgehen zur Entwicklung des konzeptuellen Modells welches als Grundlage eines entsprechenden Wireframes dient.

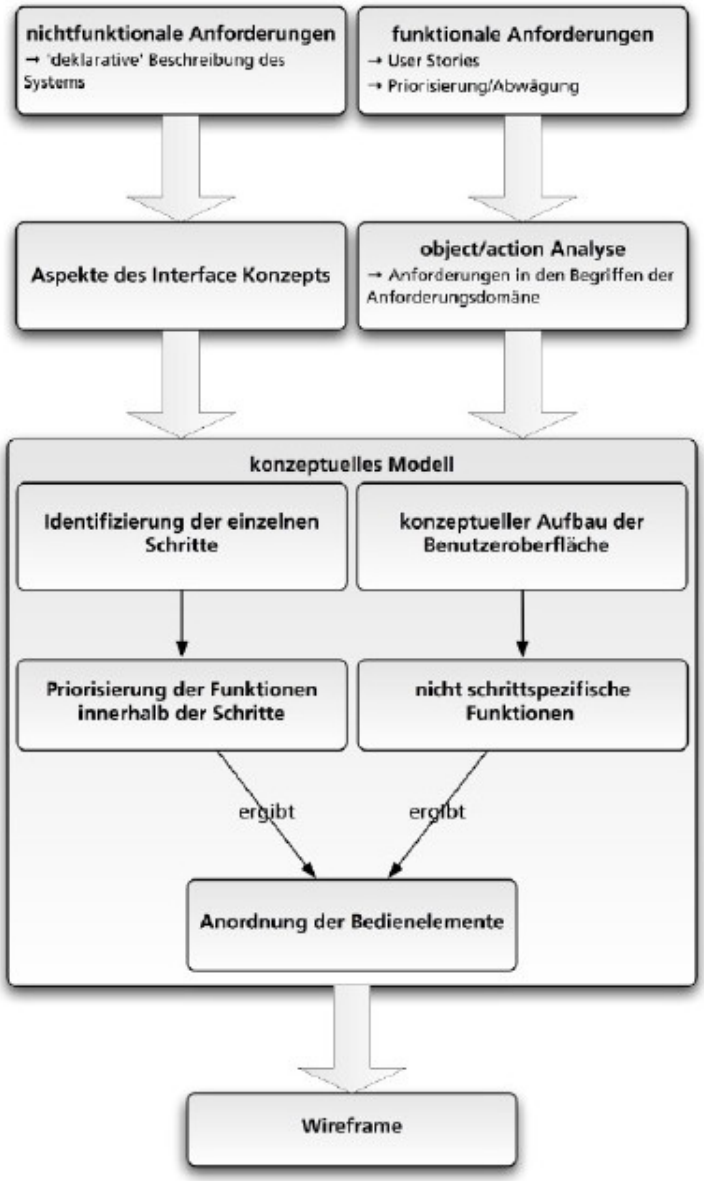


Abbildung 7.1: Vorgehen bei der Entwicklung des Wireframes

Bei dem Entwurf des konzeptuellen Modells geht es allein darum, die Anforderungen in den Begriffen der Anforderungsdomäne darzustellen. Es werden dabei die Problemstellungen der Benutzer in der Anwendungsdomäne sowie ihre Ziele, welche sie mit der Anwendung erreichen wollen, betrachtet. An dieser Stelle geht es noch nicht um das Interface, sondern um die zentralen Aspekte und Konzepte der Anwendungsdomäne: Welche Daten werden

von den Benutzern in welcher Form bearbeitet und wie lassen sich diese Daten in sinnvolle Einheiten fassen [Joh00a].

Zur Ausarbeitung des konzeptuellen Modells wurde die in [Joh00b] beschriebene Methode der object/action Analyse durchgeführt. Dabei wurden die für die Benutzer relevanten Objekte sowie die darauf auszuführenden Aktionen identifiziert.

Formular	Formularelement	E-Mail-Bestätigung	Reminder	E-Mail-Benachrichtigung
neu anlegen	Hinzufügen	Betreff setzen		Betreff setzen
in das Portal integrieren	Eigenschaften bearbeiten	Inhalt setzen		Inhalt setzen
Layout auswählen	Positionieren	Inhalt mit Formularwerten dynamisieren		(mehrere) Empfänger auswählen
ein Formular speichern	Entfernen	Versandkriterien angeben		Inhalt mit Formularwerten dynamisieren
ein bestehendes Formular öffnen	Abhängigkeiten definieren		Versanddaten angeben	Versandkriterien angeben
ein bestehendes Formular bearbeiten	Ein-/Ausblenden anhand von Kriterien			Stellvertreter angeben
	Gruppierung von Formularelementen			Dispatcher einrichten
	Formular in mehrere Seiten aufteilen			

Tabelle 7.1: Objekte/Aktionen Analyse der Anwendungsdomäne ‘einfache Prozesse in Webportalen’

Anhand dieser Identifizierung der benutzerrelevanten Objekte und Aktionen muss nun eine sinnvolle Ein- und Aufteilung der unterschiedlichen Konzepte vorgenommen werden. Da die Methode auf einem (nicht linearen) Schritt-für-Schritt Ansatz basieren soll, erfolgt zunächst eine Abgrenzung der einzelnen Schritte.

7.1 Elemente des schrittweisen Modellierungsansatzes

Zwischen denen in Tabelle 7.1 rot hervorgehobenen Objekten Formular und Formularelement besteht eine has-many-Relation. Ein Formular enthält mehrere Formularelemente, und für den Benutzer würde eine Trennung dieser beiden Objekte ein künstlich eingeführtes Konzept bedeuten, welches nicht seiner natürlichen Arbeits- und Denkweise entspricht. Solche abstrakten Ebenen sind für Benutzer nicht intuitiv, daher schwer zu verstehen und werden

leicht vergessen [Joh00b]. Entsprechend sind die Objekte Formular und Formularelemente visuell integriert in einem Schritt darzustellen.

Die Funktionen Layout auswählen sowie die Funktion Integration des Formulars in das Portal sind jeweils geschlossene und unabhängige Vorgänge. Um die Einfachheit und Übersichtlichkeit der einzelnen Schritte zu gewährleisten werden somit auch diese beiden Vorgänge jeweils in eigene Schritte ausgelagert. Dadurch bleibt in den anderen Schritten die Komplexität dieser Vorgänge verborgen, so können die einzelnen Schritte sich besser strukturiert gestalten.

Auf den Objekten E-Mail-Bestätigung und Reminder werden weitgehend die gleichen Funktionen ausgeführt. Das Reminder-Objekt ist funktional gesehen eine Erweiterung des E-Mail-Bestätigung-Objekts. Auch hier wäre eine Trennung der Konzepte E-Mail-Bestätigung und Reminder für den Benutzer nicht sofort intuitiv nachzuvollziehen. Das Konzept einer E-Mail-Bestätigung ist für die Benutzer der beschriebenen Zielgruppe klar und geläufig. Ein Reminder ist ein nicht so offensichtliches und intuitives Konzept für die Benutzer, über dessen Zweck sie erst nachdenken müssten. Eine E-Mail-Bestätigung, für die man ein Versanddatum angeben kann, entspricht weitaus mehr der natürlichen Denkweise der Benutzer, von daher wird der Reminder nicht als eigenes Objekt umgesetzt, sondern einer E-Mail-Bestätigung wird eine Option für ein Versanddatum hinzugefügt. Die Definition der E-Mail Berichte wird in einem eigenen Schritt erfolgen.

Das Festlegen der Empfänger für die Formulardaten wird repräsentiert durch das benutzerrelevante Objekt E-Mail-Benachrichtigung. Es beinhaltet Funktionen, welche ebenfalls auf dem Objekt E-Mail-Bestätigung ausgeführt werden. Jedoch wird durch dieses Objekt ein anderer Aspekt des gesamten Prozesses repräsentiert, nämlich die interne Verarbeitung der Formulardaten. Im Modellierungsprozess soll dieser Aspekt in einem dedizierten Schritt behandelt werden, um dem Benutzer die Abgrenzung der E-Mail-Benachrichtigung und der Verarbeitung der Formulardaten evident zu machen. Des Weiteren ist dieser Schritt weitaus komplexer als das Definieren der E-Mail-Bestätigungen an die Benutzer des Formulars. Von daher ist eine Trennung dieser Schritte auch wieder aus Gründen der Übersichtlichkeit und der Komplexitätsreduktion pro Schritt sinnvoll.

7.2 Reihenfolge der identifizierten Schritte

Die identifizierten Einzelschritte des Modellierungsprozesses müssen nun in eine für den Benutzer sinnvolle Reihenfolge gebracht werden. Dabei ist festzuhalten, dass die Reihenfolge der Bearbeitung der unterschiedlichen Schritte nicht als zwingend linear zu betrachten ist. Wie die Psychologie des Programmierens in [Gre96] hinsichtlich der Reihenfolge der Programmentwicklung besagt, erfolgt die Entwicklung eines Programms nicht linear und auch

nicht vom höchsten mentalen Konstrukt zum kleinsten herunter. Daraus wird in [Gre96] gefolgert, dass eine Anwendung die Reihenfolge der Modellierung beim Benutzer belassen muss und diese nicht selbst vorgeben darf. Entsprechend repräsentiert die Anordnung der einzelnen Schritte lediglich eine Möglichkeit der Abarbeitung, es muss jedoch die Möglichkeit bestehen, zwischen diesen beliebig zu navigieren.

Der erste Schritt im Modellierungsprozess muss in der Auswahl des Designs und Layouts für das Formular bestehen. Technisch gesehen ist es nicht notwendig diesen Schritt an erster Stelle zu platzieren, es könnten ohne weiteres zuerst die Elemente des Formulars oder andere Funktionalität definiert werden. Für die Zielgruppe der Anwendung, welche eben nicht über technisches Fachwissen der Web-Programmierung verfügt, ist die Trennung von Design und Funktionalität kein geläufiges Konzept. Es entspricht nicht der natürlichen Denkweise des Benutzers, vorerst vom Design zu abstrahieren und sich zunächst um die Funktionalität zu kümmern. In nahezu allen geläufigen Textverarbeitungs- oder Präsentationsanwendungen wird dem Benutzer zuerst die Auswahl einer Vorlage oder eines Designs angeboten. Beispielhaft ist dieses in Abbildung 7.2 gezeigt, welche den Schritt der Vorlagenauswahl bei dem Textverarbeitungsprogramm *Pages* zeigt, welches Teil der *Apple* Office-Suite *iWork* ist. Basierend auf dem gewählten Design arbeitet der Benutzer dann weiter und sieht sofort, wie seine Bearbeitung in dem gewählten Design aussieht.

Aus dieser Vorgehensweise leitet sich die Anordnung der Formularbearbeitung als zweiter Schritt her. Der Benutzer möchte nun die konkreten Formularelemente in das gewählte Layout einfügen. Bei den weiteren Schritten ist es sinnvoll, wenn der Benutzer das Formular visuell präsent hat, anstatt bei der Durchführung der weiteren Schritte auf einem abstrakten Objekt seiner Vorstellung zu operieren. Auch ist es technisch gesehen für die weiteren Schritte von Vorteil, wenn der Benutzer schon möglichst alle Formularelemente hinzugefügt hat. Beim Definieren von Versandkriterien oder dem Dynamisieren der E-Mail Inhalte werden mögliche Formularelemente vom System zur Auswahl angeboten. Es ist somit besser, wenn der Benutzer in diesen Schritten bereits die benötigten Formularelemente definiert hat. Ansonsten muss der Benutzer durch das System zur Formularbearbeitung zurück geschickt werden, mit dem Hinweis, dass er zuerst bestimmte Elemente dem Formular hinzufügen muss.

Die nächsten beiden Schritte bestehen im Definieren und Aktivieren der E-Mails an den Benutzer des Formulars und im Festlegen der internen Verarbeitung der Formulardaten. Technisch gesehen ist es irrelevant, welcher der beiden Schritte zuerst kommt. Die beiden Schritte sind konzeptionell identisch aufgebaut und umfassen zum Teil die gleichen Funktionen. Der Schritt für die Verarbeitung der Formulardaten ist jedoch von einem größeren Komplexitätsgrad als der Schritt zum Definieren der E-Mail-Bestätigungen an den Benutzer des Formulars. Daher ist es sinnvoll, diesen Schritt zuerst durchzuführen, damit der Benutzer dann im vom Umfang her komplexeren Schritt mit schon bekannten Konzepten konfrontiert wird.

Der Schritt zur Aktivierung, also die Einbindung des Formulars in das Portal, muss als letzter Schritt durchgeführt werden, da die gültige Ausführung der anderen Schritte Voraussetzung

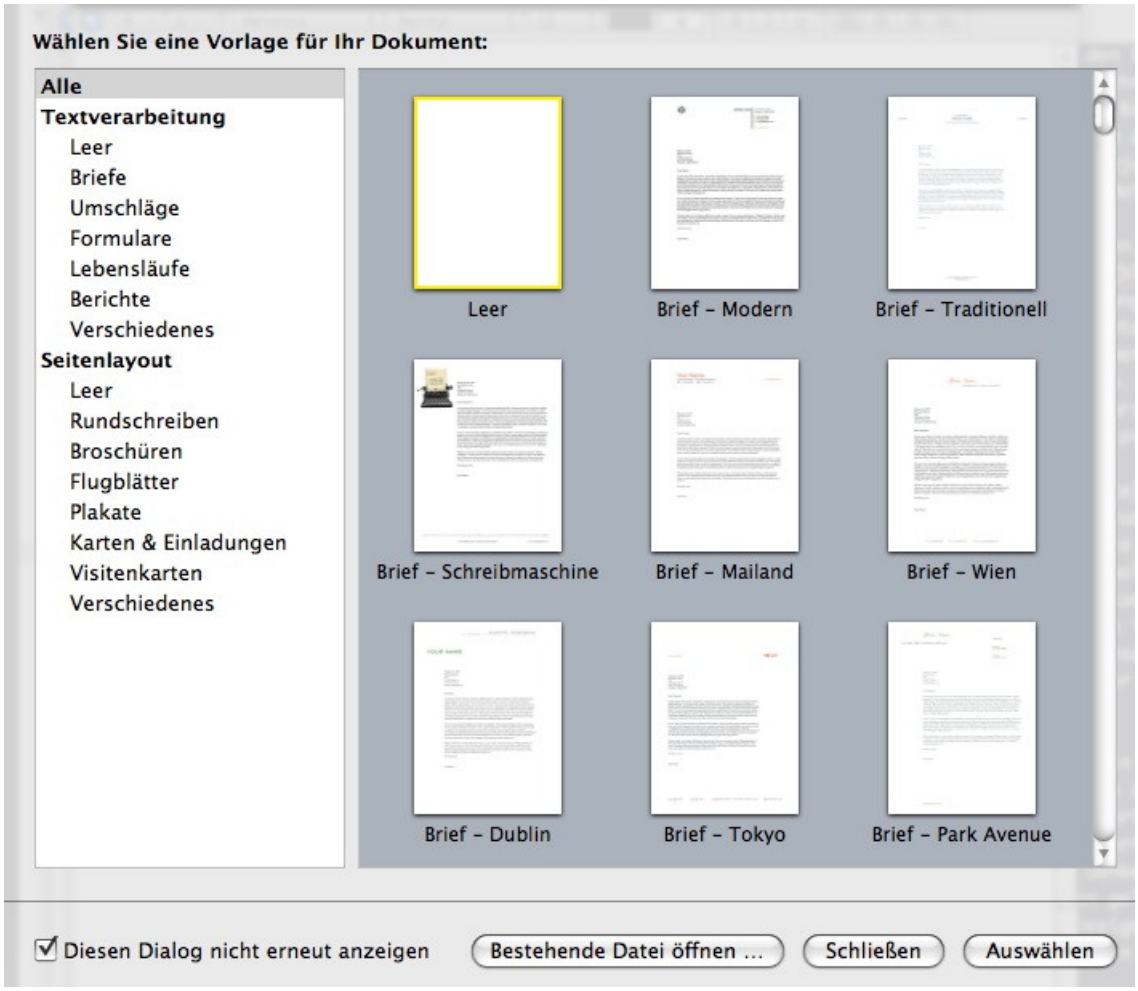


Abbildung 7.2: Auswahl einer Dokumentenvorlage in dem Textverarbeitungsprogramm *Pages*

für diesen Schritt ist.

7.3 Darstellung des schrittweisen Modellierungsansatzes

Nachdem die einzelnen Schritte und deren Reihenfolge identifiziert sind, ist es wichtig, diese Schritte und die Empfehlung der Reihenfolge in geeigneter Weise darzustellen. Dabei ist zu beachten, dass diese Darstellung

- dem Benutzer die zu durchlaufenden Schritte übersichtlich darstellt
- den Benutzer durch die einzelnen Schritte des Modellierungsprozesses führt

- dem Benutzer eine Übersicht gibt, an welcher Stelle des Modellierungsprozesses er sich gerade befindet
- dem Benutzer die optimale Reihenfolge der Schritte empfiehlt
- dem Benutzter deutlich macht, dass auch eine Navigation durch die einzelnen Schritte möglich ist

Im Folgenden wurden verschiedene Ansätze zur Darstellung des Modellierungsprozesses im Dialog mit den Benutzern diskutiert und entwickelt. Die folgenden gezeigten Abbildungen stellen einen Ausschnitt des Benutzerinterfaces dar. Im oberen Teil der Abbildungen ist das Menü enthalten, welches schrittübergreifende Funktionen wie das Speichern, Öffnen oder Neuanlegen eines Formulars bereitstellt. Darunter ist die Darstellung der Prozessschritte angeordnet, welche oberhalb des Elements liegt, das die schrittspezifischen Interaktionselemente enthält. Zur Unterstützung der Navigationsmöglichkeit wurden zwei Bedienelemente zur schrittweisen Vor- und Zurück-Navigation hinzugefügt. Desweiteren ist zu bedenken, dass zum Hervorheben der Navigationsmöglichkeit in den gezeigten Ansätzen visuelle Hilfsmittel wie Hover-Effekte zum Einsatz kommen sollten.

Die ersten beiden Ansätze in Abbildung 7.3 und 7.4 stellen die einzelnen Modellierungsschritte durch verbundene Steuerelemente dar. Diese Art der Visualisierung von Prozessschritten ist vielen Benutzern aus Kontexten wie z.B. der Darstellung von Bestellvorgängen in Online-Shops geläufig und bietet eine sehr gute Orientierung bzgl. der durchzuführenden Schritte und der aktuellen Position im Vorgang. Im Kontext von Bestellvorgängen dient dieser Ansatz häufig der reinen Visualisierung des Prozessfortschritts, bietet jedoch keine Navigationsmöglichkeit. Diese kann oft nur schrittweise über gesonderte Schaltflächen erfolgen. Auch wird in diesen beiden ersten Ansätzen die empfohlene Reihenfolge lediglich durch die Anordnung der Interaktionselemente angedeutet, es wird jedoch nicht das Voranschreiten in einem Prozessablauf visualisiert.

In Abbildung 7.3 sind die einzelnen Schaltflächen als Buttons dargestellt. Dieses hat zum einen den Vorteil, dass Buttons von den Benutzern als Interaktionselemente wahrgenommen werden und somit diese Darstellung dem Benutzer eine Interaktionsmöglichkeit nahelegt. Andererseits ist ein Button ein Steuerelement und steht im Allgemeinen für den Abschluss eines Vorgangs und den Start des nächsten Vorgangs bzw. das Auslösen einer Funktion. Um dem Benutzer die Navigationsmöglichkeit innerhalb des Modellierungsprozesses zu verdeutlichen, ist eine Verwendung von Steuerlementen daher nicht von Vorteil.

Abbildung 7.4 zeigt ebenfalls den Ansatz der verbundenen Interaktionselemente, jedoch sind die Interaktionselemente hier nicht als Steuerelemente dargestellt, sondern als rechteckige Interaktionselemente. Damit soll für den Benutzer zum einen die Möglichkeit der Interaktion hervorgehoben werden, gleichzeitig aber die Vorstellung genommen werden, dass durch das Betätigen dieser Interaktionselemente eine Funktion ausgelöst wird. Zusätzlich zur Anordnung der Interaktionselemente wird dem Benutzer durch Farbcodes visualisiert, welche

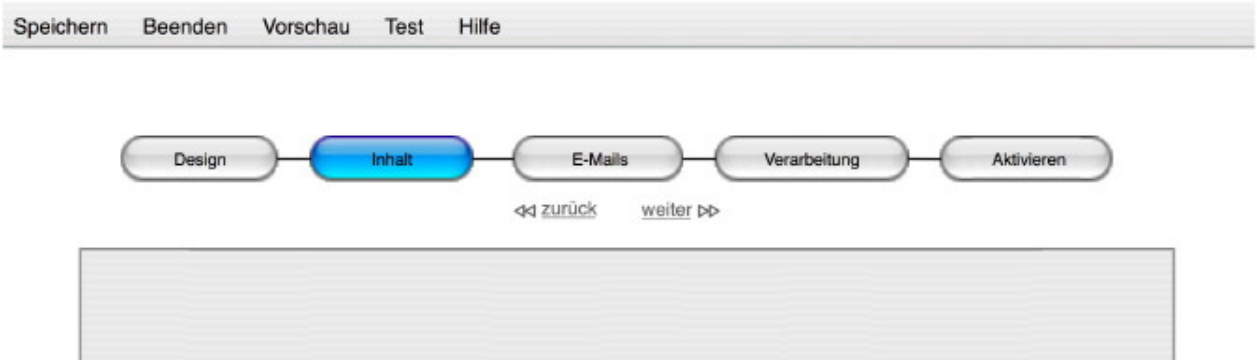


Abbildung 7.3: Darstellung der Modellierungsschritte durch verbundene Steuerelemente

Schritte er bereits erfolgreich durchlaufen hat (grün hinterlegt), welchen Schritt er gerade bearbeitet (hellblau hinterlegt) und welche Schritte er noch nicht bearbeitet hat (rot hinterlegt). Diese Farbgebung wurde von den Benutzern jedoch nicht als sofort verständlich empfunden und führte zu Verwirrung.

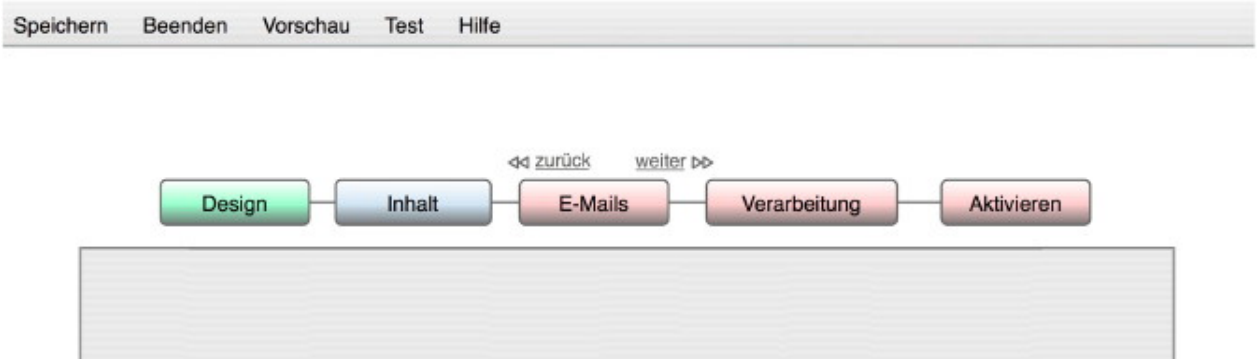


Abbildung 7.4: Darstellung der Modellierungsschritte durch farbige, verbundene Interaktionselemente

Ein weiterer, in Abbildung 7.5 illustrierter Ansatz besteht darin, die einzelnen Modellierungsschritte als Registerkarten mit entsprechenden Reitern darzustellen. Registerkarten dienen dazu, zusammengehörige Bedienelemente zu gruppieren und die Komplexität der Benutzeroberfläche zu reduzieren, indem Optionen oder Informationen auf mehrere, übereinander liegende Dialoge verteilt werden. Dieses ist im Prinzip im Kontext der einzelnen Modellierungsschritte gefordert, jedoch können die Reiter der einzelnen Registerkarten keine übersichtliche Darstellung eines Gesamtprozesses leisten. Zwar verdeutlichen die Reiter der Registerkarten sehr gut, dass eine beliebige Navigation zwischen den Schritten möglich ist, aber der Benutzer wird nicht in der Reihenfolge der Bearbeitung geleitet und es wird auch nicht herausgestellt, dass jeder der angegebenen Schritte durchlaufen werden muss.

Abbildung 7.5: Darstellung der Modellierungsschritte durch Registerkarten

Mit dem in Abbildung 7.6 gezeigten Ansatz sind die Interaktionselemente, welche die einzelnen Schritte im Modellierungsprozess repräsentieren, in Anlehnung an ein Menü gestaltet und angeordnet, wodurch die Navigationsmöglichkeit auf diesen einzelnen Schritten verdeutlicht wird.

Abbildung 7.6: Darstellung der Modellierungsschritte mit Zahlen

Durch die dazwischen angeordneten Pfeile wird das schrittweise Voranschreiten im Modellierungsprozess skizziert, wobei durch die Nummerierung der einzelnen Schritte zusätzlich die Empfehlung der Reihenfolge gegeben wird. Es werden nur sehr begrenzt und leichtgewichtig Farbcodes eingesetzt: es wird der aktive Schritt gegenüber den anderen vorhandenen Schritten farblich hervorgehoben und nicht ausführbare Schritte werden als nicht auswählbar gekennzeichnet. Diese Art der Kennzeichnung war für die Benutzer weniger verwirrend als die in Abbildung 7.4 verwendeten dominanten und komplexeren Farbcodes. Die einzelnen Schritte sind sehr übersichtlich dargestellt und der Benutzer hat durch die Hervorhebung des aktuellen Bearbeitungsschrittes eine gute Orientierung im Modellierungsprozess. Problematisch bei diesem Ansatz ist, dass es sich hier um eine ungewöhnliche und nicht geläufige Darstellung eines Prozesses handelt. Bei den zwischen den Interaktionselementen der einzelnen Schritte eingefügten Pfeilen war den Benutzern nicht klar, ob diese Interaktions-

elemente sind oder ob es sich hierbei um funktionslose Designelemente handelt. Dadurch bestand bei den Benutzern Unklarheit über die Bedeutung und Bedienung der gezeigten Interaktionselemente.

Um die einzelnen Schritte als fortschreitenden Prozess richtungsweisend darzustellen, sind Pfeile ein sehr leicht verständliches und eindeutiges Hilfsmittel, wenn sie in nicht irritierender Form eingesetzt werden. In Abbildung 7.8 sind die Interaktionselemente wie in Abbildung 7.6 wieder wie Menüelemente angeordnet, was die Interaktions- bzw. Navigationsmöglichkeit hervorhebt. Die Darstellung der einzelnen Interaktionselemente in pfeilartiger Form unterstreicht sehr gut die lineare Abfolge der Schritte, ohne, wie im vorherigen Ansatz, zusätzliche Designelemente hinzuzufügen.

Des Weiteren entsprechen die Interaktionselemente durch Form und Anordnung einer Breadcrumb-Navigation, welche für Benutzer ein geläufiges Konzept zur Orientierung in komplexeren Konzepten darstellt. Breadcrumb-Navigationen führen bei Benutzern zu keinerlei Problemen, da sie normalerweise kleine Bedienelemente sind, welche oft nicht wahrgenommen werden oder, wenn sie wahrgenommen werden, sie in ihrer Bedeutung und Benutzung problemlos verstanden werden [bre]. Beispielhaft ist in Abbildung 7.7 die Breadcrumb-Navigation des Internet-Portals der Freien Universität Berlin gezeigt.

Abbildung 7.7: Breadcrumb-Navigation des Internetportals der Freien Universität Berlin

Aufgrund der Größe, Gestaltung und Anordnung der in Abbildung 7.8 gezeigten Navigation kommt der Aspekt des Nicht-Wahrnehmens durch den Benutzer an dieser Stelle nicht zum Tragen, die positive Eigenschaft der problemlosen Bedienung behält jedoch ihre Gültigkeit. Somit wird durch diese Anlehnung an eine Breadcrumb-Navigation zusätzlich die Möglichkeit der beliebigen Navigation innerhalb des Prozesses betont.

Abbildung 7.8: Darstellung der Modellierungsschritte ähnlich einer Breadcrumb-Navigation

Da die Nummerierung der Schritte im Modellierungsschritte als Unterstreichung der empfohlenen Reihenfolge in Abbildung 7.6 als positiver und erleichternder Aspekt von den Benutzern wahrgenommen wurde, wurde der Ansatz der Breadcrumb-Navigation in Abbildung 7.9 diesbezüglich erweitert.

Abbildung 7.9: Darstellung der Modellierungsschritte ähnlich einer breiten Breadcrumb-Navigation mit farbiger Nummerierung

Insgesamt wurde von den Benutzern der in Abbildung 7.9 gezeigte Ansatz zur Darstellung des Prozessablaufes als intuitivste und am besten verständliche Form der Darstellung des Prozessablaufes bewertet. Es werden jederzeit alle Schritte des Modellierungsprozesses übersichtlich dargestellt und es wird deutlich hervorgehoben, an welcher Stelle des Prozesses sich der Benutzer gerade befindet. Die Benutzer erkannten bei dieser Darstellung in Anlehnung an eine Breadcrumb-Navigation ganz klar die Interaktionselemente zur Navigation zwischen den einzelnen Prozessschritten und sie fühlten sich im Ablauf des Prozesses gut geführt, was durch die Form der Interaktionselemente und die Nummerierung der Prozessschritte unterstützt wird.

7.4 Konzeptueller Aufbau der einzelnen Schritte

7.4.1 Auswahl des Layouts

Bei der Auswahl des Layouts müssen dem Benutzer, wie bereits in Abbildung 7.2 gezeigt, in einem Dialogfenster in übersichtlicher Form zur Verfügung stehende Formularlayouts zur Auswahl angeboten werden. Dem Benutzer muss ersichtlich sein, wie bestimmte Formularelemente angeordnet und aussehen werden und welche Schriftarten, -größen und -stile verwendet werden. Damit ermöglicht man dem Benutzer möglichst früh von der abstrakten Vorstellung des Formulars und dessen Aussehen zu einem konkreten und visuell greifbaren Formular zu kommen.

Auch ist es empfehlenswert, in diesem Schritt bereits Formularvorlagen für bestimmte Einsatzzwecke wie z.B. eine Kontaktanfrage anzubieten, welche schon entsprechende Formularelemente enthalten. Der Benutzer kann diese dann beliebig bearbeiten und erweitern.

7.4.2 Bearbeitung des Formulars und der Formularelemente

Die Bearbeitung des Formulars stellt den zentralen Schritt des Modellierungsprozesses dar. Die durch den Benutzer ausgeführten Hauptfunktionen in diesem Schritt sind das Hinzufügen von Formularelementen zum Formular sowie die Bearbeitung der Eigenschaften der hinzugefügten Formularelemente.

Zum konzeptionellen Aufbau dieses Schrittes wurden drei Ansätze herausgearbeitet, die sich hinsichtlich der Anordnung ihrer Bedienelemente unterscheiden. Alle drei Ansätzen basieren auf dem Drag-and-Drop Bedienungskonzept und der WYSIWYG-Darstellung des Formulars. Ein Vorteil des Drag-and-Drop Bedienungskonzeptes ist neben der Einfachheit der Aspekt, dass die Elemente im Formular unmittelbar positioniert werden können, es also nicht einer weiteren Einstellungsoption zur Positionierung der Elemente bedarf.

Es hat sich in den Usability-Tests herausgestellt, dass zu viele Bedienelemente dazu führen, dass die Benutzer diese nicht wahrnehmen und die gebotenen Funktionen nicht im Überblick haben können. Daher wird auf Bedienelemente links und unterhalb des Formulars verzichtet.

Aufgrund der Forderung, dass die Benutzer zum Bearbeiten der Formularelemente nicht den Fokus des Formulars verlassen möchten, erfolgt die Bearbeitung eines Formularelements nicht in einem separaten Fenster oder Dialog, sondern direkt im Eigenschaften-Inspektor. Der Benutzer erhält so in der WYSIWYG Ansicht des Formulars sofort das visuelle Feedback zu seinen vorgenommenen Einstellungen und hat jederzeit einen Überblick über das gesamte Formular. Der Eigenschaften-Inspektor zeigt die Eigenschaften eines ausgewählten Elements an und bietet die Möglichkeit der Bearbeitung. Innerhalb des Eigenschaften-

Inspektors werden die Bedienelemente nach bearbeitbar und nicht-bearbeitbar strukturiert und deutlich als solche gekennzeichnet. Insgesamt wurde von den Benutzern nach Wahrnehmung des Eigenschaften-Inspektors recht schnell erkannt, dass dieser auch der Bearbeitung der Eigenschaften dient, da dieses ein vertrautes und verbreitetes Konzept im Umgang mit Eigenschaften-Inspektoren ist.

Durch den Einsatz des Drag-and-Drop Bedienungskonzepts ist es aus Gründen der Softwareergonomie und des Bedienungskomforts erforderlich, dass das Bedienelement, welches die Drag-Zone darstellt (hier: die Formularelemente bereitstellt) relativ nahe bei der Drop-Zone (dem Formular) angeordnet sein muss.

Insgesamt ergibt sich die folgende Priorisierung der in diesem Schritt zu realisierenden Funktionen:

1 Hinzufügen von Formularelementen

2 Bearbeiten der Eigenschaften der Formularelemente

3 Positionieren eines Formularelements

4 Entfernen eines Formularelements aus dem Formular

5 Definieren von Abhängigkeiten zwischen Formularelementen

6 Ein- und Ausblenden von Formularelementen anhand von Kriterien

7 Gruppieren von Formularelementen

8 Aufteilen des Formulars in mehrere Seiten

Der erste Ansatz entspricht der Anordnung der Bedienelemente wie in der Anwendung *jotform*, welche Gegenstand der Usability-Tests war und ebenfalls auf dem Drag-and-Drop Bedienungskonzept basiert. Damit ergibt sich die in Abbildung 7.10 gezeigte Anordnung der Bedienelemente innerhalb des Schrittes zur Formularbearbeitung. Das Bedienelement mit den einzufügenden Formularelementen ist rechts neben dem Formular oberhalb des Eigenschaften-Inspektors angeordnet.

Bei dieser Anordnung haben die Usability-Tests gezeigt, dass der Eigenschaften-Inspektor aufgrund seiner Positionierung nur schwer wahrgenommen wird. Um die Wahrnehmung des Eigenschaften-Inspektors zu verbessern, könnte er so lange mit einem Hinweistext belegt sein, wie kein Formularelement ausgewählt ist. Des Weiteren ergeben sich für den Benutzer bei der Bearbeitung lange Mauswege und relativ weite Blickbewegungen. Dies ist unter dem Aspekt, dass das Bearbeiten der Formularelemente eine der Hauptfunktionen in diesem Bearbeitungsschritt darstellt, ungünstig.

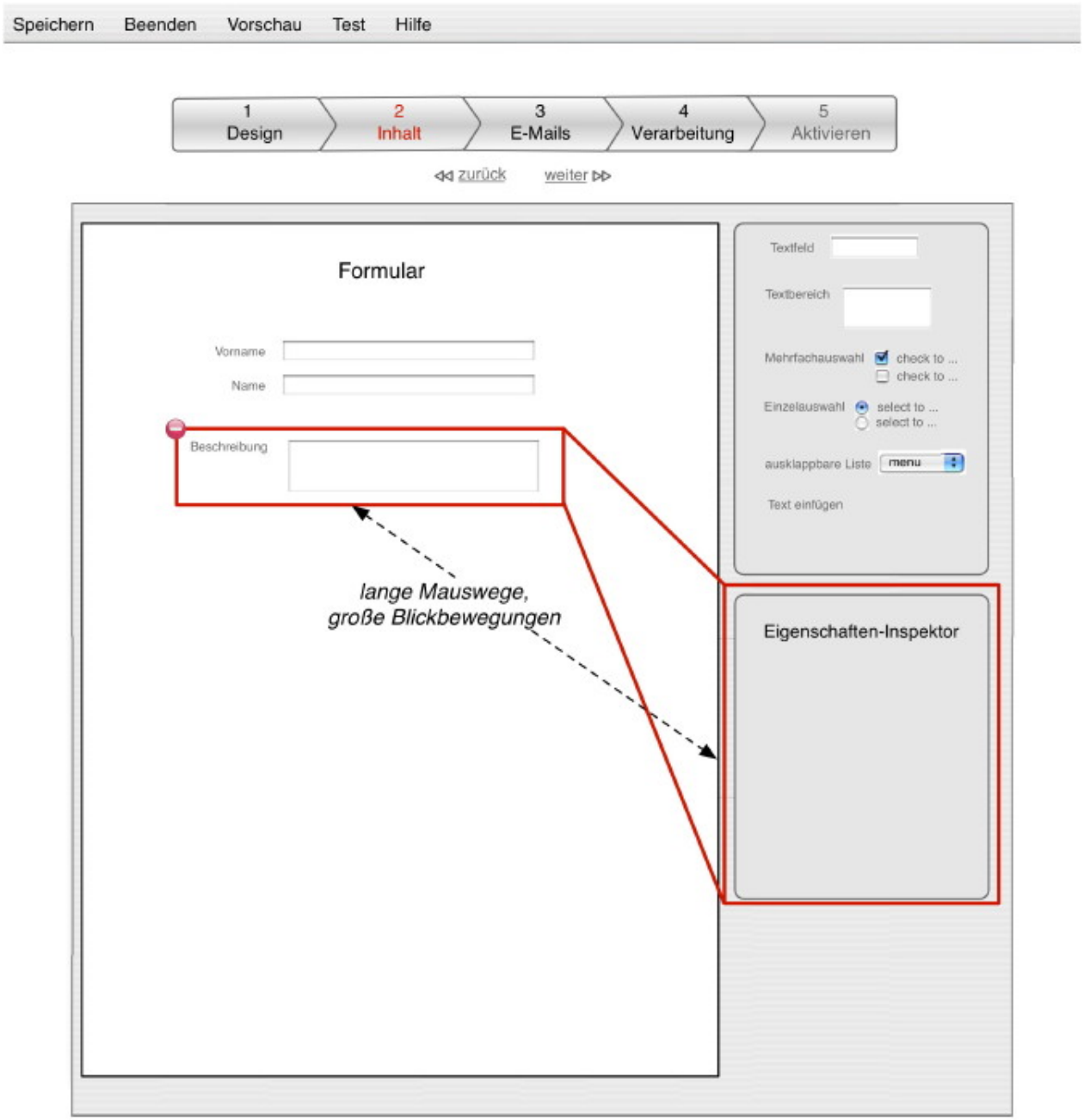


Abbildung 7.10: Anordnung des Eigenschaften-Inspektors unten rechts

Der zweite Ansatz führt zu einer besseren Wahrnehmung des Eigenschaften-Inspektors, da dieser, wie in Abbildung 7.11 gezeigt, nun direkt oben rechts neben dem Formular angeordnet ist und somit besser in den Fokus des Benutzers fällt. Auch werden die langen Mauswege und großen Blickbewegungen bei der Bearbeitung der Formularelemente vermieden.

Durch die Anordnung des Bedienelements mit den einzufügenden Formularelementen oberhalb der WYSIWYG-Darstellung des Formulars, bekommt dieses Element den Charakter einer Toolbar, was ein geläufiges Konzept für Benutzer ist. Wie durch die gestrichelten Pfeile dargestellt, entstehen auch beim Einfügen von Formularelementen keine langen Mauswege oder notwendige große Blickbewegungen für den Benutzer. Jedoch wird der sichtbare Bereich des Formulars deutlich reduziert. Wie im Abschnitt 2.2.1 (Psychologie der Programmierung) unter dem Stichpunkt 'Effekt der Umgebung' beschrieben, macht ein Editor, der einen einfachen Zugang zu einem großen Fenster/Bereich an Code ermöglicht, den Entwicklungszyklus erheblich einfacher. Der Programmierer muss lesen und verstehen können, was er bisher entwickelt hat, um seine neuen Informationen/Gedanken damit zu verbinden. Das 'window of access' sollte möglichst groß sein, was bei diesem Ansatz nicht gegeben ist.

Ein positiver Aspekt der in Abbildung 7.11 gezeigten Anordnung besteht darin, dass die Möglichkeit besteht, unterhalb des Eigenschaften-Inspektors ein Element mit weiterführenden Informationen oder Hilfetexten zu platzieren.

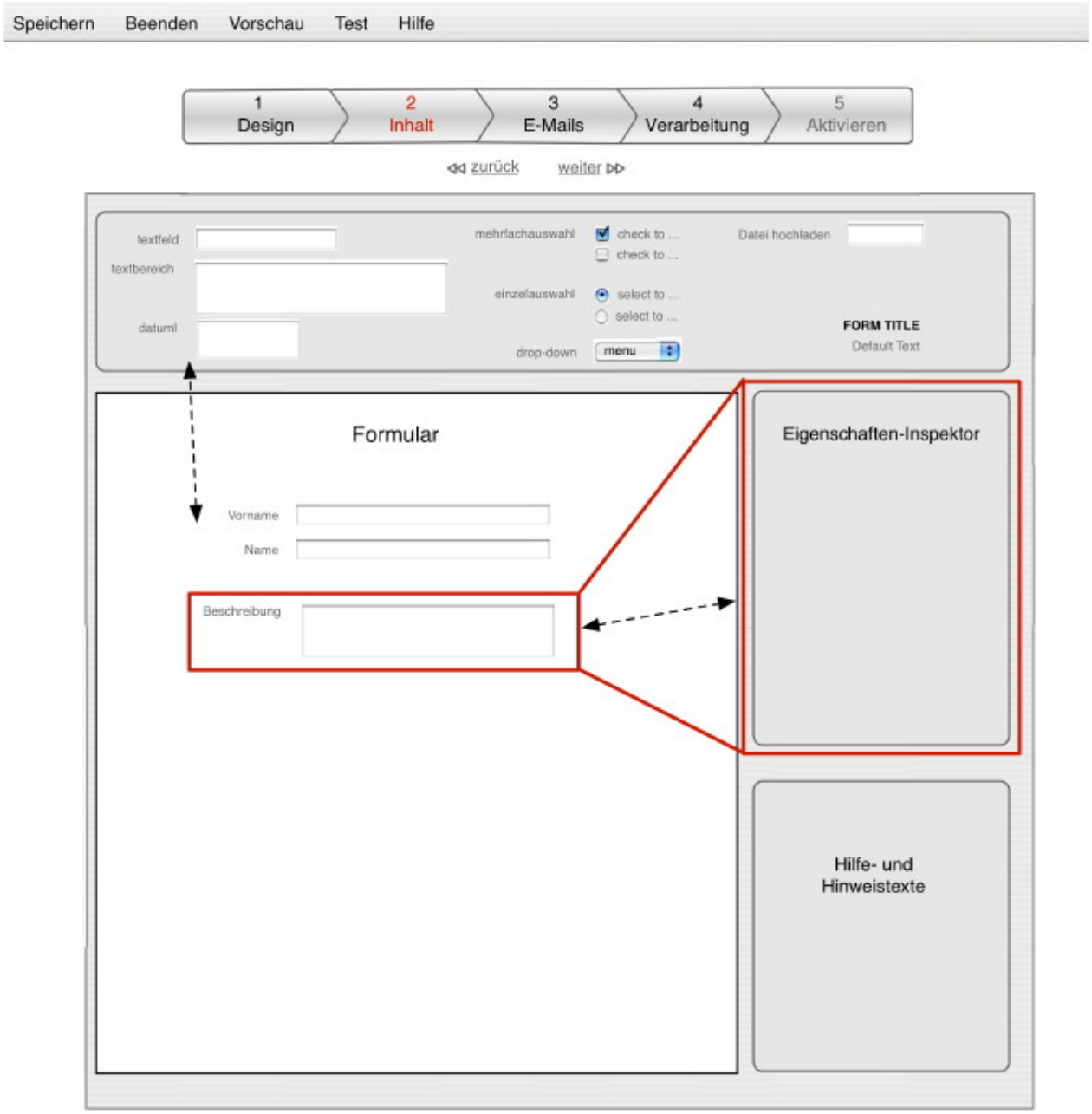


Abbildung 7.11: Anordnung des Eigenschaften-Inspektors oben rechts

Im dritten diskutierten Ansatz in Abbildung 7.12 werden die Probleme, die sich in der Diskussion der ersten beiden Ansätze gezeigt haben, insofern behoben, als dass rechts neben dem Formular je nach Kontext nur die einzufügenden Formularelemente oder der Eigenschaften-Inspektor angezeigt werden.

Diese beiden Bedienelemente beinhalten Funktionen, die jeweils nur in einem der beiden Bearbeitungsmodi, dem Hinzufügen von Elementen zum Formular bzw. dem Bearbeiten der Eigenschaften der Formularelemente, verfügbar sein müssen. Im Bearbeitungsmodus 'Hinzufügen von Elementen' ist das Bedienelement mit den einzufügenden Formularelementen sichtbar und im Modus 'Bearbeiten von Formularelementen' ist der Eigenschaften-Inspektor sichtbar.

Dies hat den Vorteil, dass der Benutzer in seinem jeweiligen Bearbeitungsmodus nur mit den relevanten Optionen konfrontiert wird, aber trotzdem eine Übersicht aller vorhandenen Bedienelemente hat. In keinem der Bearbeitungsmodi wird der Fokus des Formulars verlassen, und aufgrund der Anordnung der Bedienelemente entstehen keine langen Mauswege und keine unvermeidbaren großen Blickbewegungen für den Benutzer. Hinzu kommt, dass die Höhe des Formulars durch die Anordnung der Bedienelemente nicht beeinflusst wird.

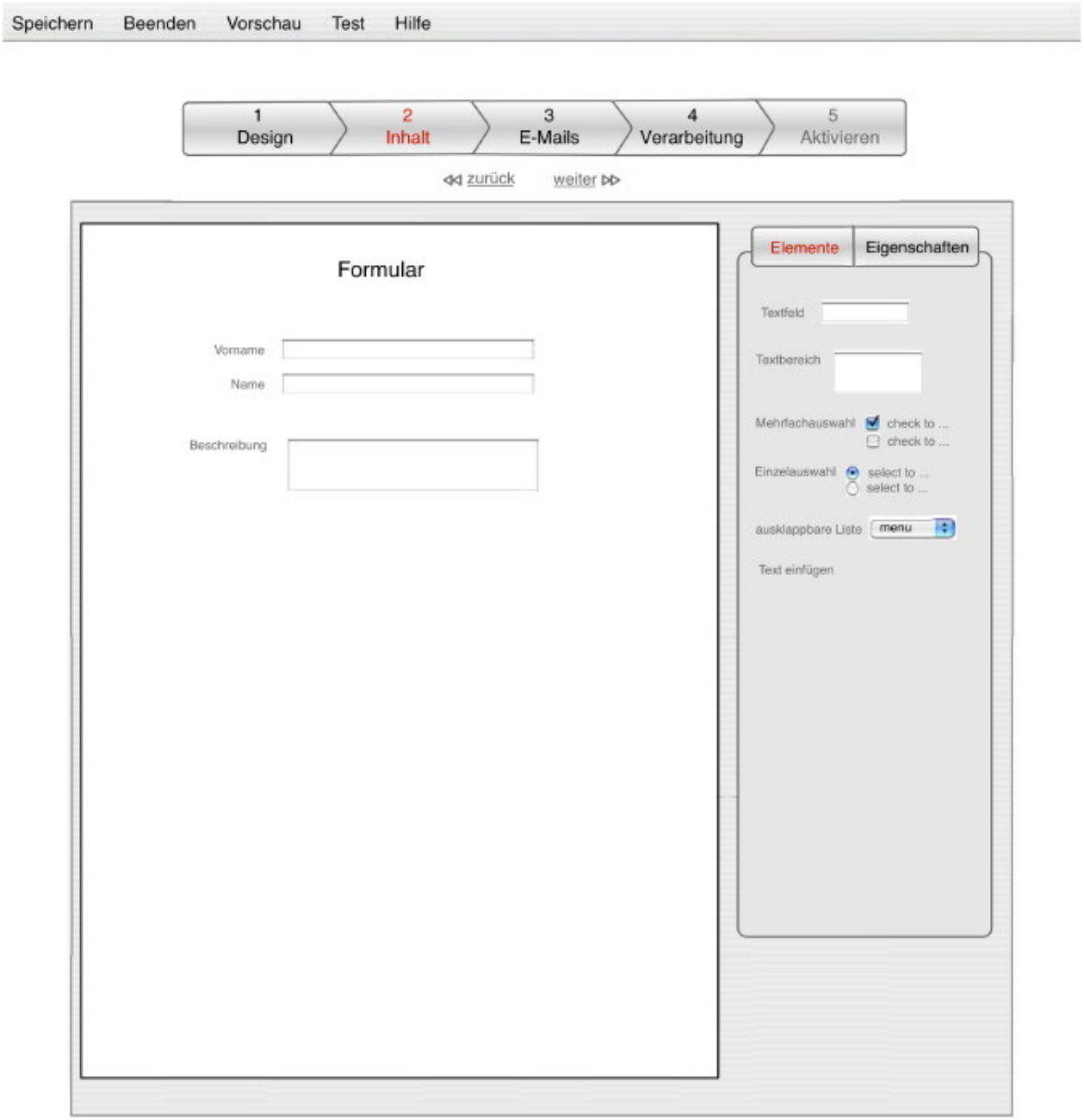


Abbildung 7.12: Wechselnde Anzeige von Eigenschaften-Inspektor und Formularelementen

7.4.2.1 Untersuchungen zum Bedienkonzept unter Berücksichtigung verschiedener Benutzertypen

Bei der Umsetzung des im vorhergehenden Abschnitt beschriebenen Oberflächenkonzepts mit wechselnden Bedienelementen für die beiden Bearbeitungsmodi ist das Verhalten zweier unterschiedlicher Benutzertypen zu berücksichtigen.

Zum einen gibt es den Benutzertyp, der zunächst nacheinander alle Elemente dem Formular hinzufügt und anschließend die Eigenschaften aller hinzugefügten Formularelemente nacheinander bearbeitet. Dieser Benutzertyp wird im Folgenden als kontinuierlicher Benutzertyp bezeichnet. Der zweite Benutzertyp fügt ein Element ein und bearbeitet zunächst dessen Eigenschaften, bevor er das nächste Element dem Formular hinzufügt. Dieser Benutzertyp alterniert also zwischen den beiden Bearbeitungsmodi und wird im Folgenden daher als alternierender Benutzertyp bezeichnet.

Generell soll es so sein, dass der Eigenschaften-Inspektor die Eigenschaften eines aktiven, also ausgewählten Formularelements anzeigt. Entscheidende Punkte sind an dieser Stelle, ob ein Formularelement nach dem Einfügen in das Formular automatisch aktiv ist oder nicht und ob der Eigenschaften-Inspektor bei der Aktivierung eines Formularelements automatisch in den Vordergrund tritt oder nicht.

Um das optimale Verhalten der Anwendung bzgl. der Anzeige der Bedienelemente und der Aktivierung von Formularelementen zu entwickeln, werden im Folgenden die notwendigen Mausklicks der unterschiedlichen Benutzertypen verglichen, welche für den Wechsel zwischen den Bearbeitungsmodi beim Einfügen und Bearbeiten von fünf Formularelementen notwendig sind. Als Voraussetzung gilt dabei, dass vor dem Einfügen des ersten Elements der Einfügemodus aktiv ist.

Für den alternierenden Benutzer ergeben sich die in Tabelle 7.2 gezeigten Mausklicks.

	Inspektor springt in den Vordergrund bei aktivem Formularelement	Inspektor springt nicht in den Vordergrund bei aktivem Formularelement
Formularelement aktiv nach Einfügen in das Formular	4 Klicks, um bei den ersten vier Formularelementen nach dem Bearbeiten wieder in den Modus zum Einfügen eines weiteren Formularelements zu kommen	9 Klicks, davon 5 um nach jedem Einfügen den Eigenschaften-Inspektor in den Vordergrund zu holen und 4 um nach der Bearbeitung der Elemente jeweils wieder in den Einfügemodus zu kommen
Formularelement nicht aktiv nach Einfügen in das Formular	9 Klicks, 5 um direkt nach dem Einfügen eines Elements dieses zu aktivieren (ohne Mausweg) und 4 Klicks, um nach dem Bearbeiten des eingefügten Elements wieder in den Bearbeitungsmodus zu kommen	14 Klicks, davon 5 Klicks um jedes eingefügte Element zu aktivieren, und weitere 5 um nach der Aktivierung eines Elements in den Bearbeitungsmdus zu kommen und weitere 4 Klicks um nach der Bearbeitung der ersten vier Elemente wieder in den Einfügemodus zu wechseln

Tabelle 7.2: Anzahl Mausklicks beim Einfügen und Bearbeiten von fünf Formularelementen für den alternierenden Benutzertyp

Tabelle 7.3 zeigt die notwendigen Mausklicks für den kontinuierlichen Benutzertyp:

	Inspektor springt in den Vordergrund bei aktivem Formularelement	Inspektor springt nicht in den Vordergrund bei aktivem Formularelement
Formularelement aktiv nach Einfügen in das Formular	4 Klicks Klicks, um nach dem Einfügen der ersten vier Elemente jeweils wieder in den Einfügemodus zu kommen	1 Klick, um nach Einfügen des fünften Formularelements den Eigenschaften-Inspektor in den Vordergrund zu holen
Formularelement nicht aktiv nach Einfügen in das Formular	1 Klick, um nach Einfügen des fünften Formularelements ein zu bearbeitendes Element zu aktivieren	2 Klicks, 1 Klick um nach dem Einfügen des fünften Elements ein zu bearbeitendes Element zu aktivieren und 1 weiterer Klick um dann den Eigenschaften-Inspektor in den Vordergrund zu holen

Tabelle 7.3: Anzahl Mausklicks beim Einfügen und Bearbeiten von fünf Formularelementen für den kontinuierlichen Benutzertyp

Tabelle 7.3 ist zu entnehmen, dass wenn ein eingefügtes Formularelement automatisch aktiviert wird und der Eigenschaften-Inspektor dabei automatisch in den Vordergrund tritt, dieses optimal der Arbeitsweise des alternierenden Benutzertyps entgegen kommt. Es hat jedoch zur Folge, dass der kontinuierliche Benutzertyp nach jedem Einfügen eines Formularelements zurück zu dem Bedienelement mit den einzufügenden Formularelementen springen muss. In diesem Fall kommen beide Benutzertypen mit der gleichen Anzahl Klicks aus, auch wenn dieses für den kontinuierlichen Benutzertyp nicht die optimale Anzahl an Klicks darstellt.

Wie in Tabelle 7.3 zu erkennen ist, entspricht ein Eigenschaften-Inspektor, der nicht automatisch in den Vordergrund tritt, der Arbeitsweise des kontinuierlichen Benutzertyps. Ist ein Formularelement nach dem Hinzufügen zum Formular nicht automatisch aktiviert, so wirkt sich auch ein Eigenschaften-Inspektor, der bei einem aktiven Element automatisch in den Vordergrund tritt, nicht negativ auf das Verhalten des kontinuierlichen Benutzertyps aus. Bei diesen beiden Verhaltensarten wird der kontinuierliche Benutzertyp nicht von der Anwendung gezwungen, zwischen den Bearbeitungsmodi zu wechseln, sondern er kann nacheinander seine Elemente dem Formular hinzufügen, ohne übermäßige Mauswege oder Klicks vornehmen zu müssen.

Für den alternierenden Benutzertyp wirkt sich ein Eigenschaften-Inspektor, der bei Aktivierung eines Elements nicht in den Vordergrund tritt, extrem negativ auf die Anzahl der notwendigen Klicks während seiner Bearbeitung aus, da er immer manuell in das Bedienungselement zum Bearbeiten der Eigenschaften wechseln muss.

Ist ein Formularelement nach dem Einfügen in das Formular nicht aktiviert, aber der Eigenschaften-Inspektor tritt in den Vordergrund, sobald ein Formularelement aktiv ist, so wird der kontinuierliche Benutzer in seiner Bearbeitung dadurch nicht negativ beeinflusst. Der alternierende Benutzertyp könnte bei diesem Verhalten nach Einfügen eines Elements durch einen Klick auf das eingefügte Element, welcher nicht mit Mauswegen verbunden ist, dieses aktivieren und so in den Bearbeitungsmodus wechseln. Als Konsequenz dessen wird dieses geschilderte Verhalten der Anwendung als Optimum für die Zufriedenstellung beider Benutzertypen angesehen.

7.4.3 Bearbeitung der E-Mail-Berichte und der Formulardaten

Die beiden Schritte der Erstellung der E-Mail-Berichte und der Verarbeitung der Formulardaten werden aufgrund ihrer konzeptionellen und funktionalen Überschneidungen identisch aufgebaut sein, um auch den Benutzer mit möglichst wenigen unterschiedlichen Konzepten zu konfrontieren.

Die zentralen Funktionen in diesen Schritten sind das Verfassen des Betreffs und der Inhalte für die zu verschickenden E-Mails. Gegebenenfalls wollen die Benutzer für die E-Mail-Bestätigungen noch Versandkriterien und Versanddaten angeben. In vielen Fällen wird es jedoch so sein, dass direkt nachdem das Formular abgeschickt wurde, eine Bestätigung an den Formular-Benutzer' verschickt werden soll. Dieses wird als default-Fall angesehen und entsprechend einfach wird dieser Anwendungsfall für den Benutzer aufbereitet sein. Aus Gründen der Komplexitätsreduktion in der Oberfläche wird nur ein Benutzer, der erweiterte Einstellungen für Versandkriterien oder den Versandzeitpunkt vornehmen möchte, überhaupt mit diesen Funktionen konfrontiert.

Es hat sich in den Usability-Test herausgestellt, dass die Benutzer sehr gut mit dem Konzept der Regeldefinition durch ausklappbare Listen zurecht kommen. Dieses ist ein bekanntes und verstandenes Vorgehen, beispielsweise bei der Regeldefinition in E-Mail Programmen, wie in Abbildung 7.13 illustriert. Des Weiteren bietet dieses Vorgehen sehr gut die Möglichkeit dem Benutzer die vorhandenen Optionen in den Listen aufzuzeigen und durch diese Vorgaben Falscheingaben des Benutzers zu vermeiden.

Abbildung 7.13: Dialog für das Hinzufügen von Regeln in dem E-Mail Client *Apple Mail*

Für den Schritt des Erstellens der E-Mail-Bestätigung an den Benutzer des Formulars ergibt sich die folgende Priorisierung der zu realisierenden Funktionen:

1 Inhalt setzen

2 Betreff setzen

3 Inhalt mit Formularwerten dynamisieren

4 Versandkriterien für die E-Mail definieren

5 Versanddatum für die E-Mail angeben

Für die Verarbeitung der Formulardaten ergibt sich die folgende Priorisierung der zu realisierenden Funktionen:

1 Inhalt setzen

2 Betreff setzen

3 (mehrere) Empfänger auswählen, bzw. angeben

4 Inhalt mit Formularwerten dynamisieren

5 Versandkriterien definieren

6 Stellvertreter angeben

7 Dispatcher einrichten

Zur Repräsentation der beiden genannten Schritte werden zwei Möglichkeiten aufgezeigt, welche sich in der Anordnung der zentralen Funktionen unterscheiden. Im dritten Schritt, dem Verfassen der E-Mail-Bestätigungen, sind die drei zentralen Funktionen das Verfassen der E-Mail, das Definieren der Versandkriterien sowie die Angabe eines Versandzeitpunktes. Der vierte Schritt, welcher die Verarbeitung der Formulardaten betrifft, wird nach den folgenden Funktionen aufgegliedert: Verfassen der Inhalte der E-Mails zur Verarbeitung der Formulardaten, Angeben der Empfänger und Stellvertreter, Definieren der Kriterien und dem Einrichten des Dispatchers.

Im ersten, in Abbildung 7.14 und 7.15 gezeigten Ansatz, sind alle zentralen Funktionen dieses Schrittes untereinander in einem Dialogfenster angeordnet. Neben diesem Dialogfenster hat der Benutzer jederzeit eine Übersicht aller vorhandenen E-Mail-Bestätigungen und kann diese auch einfach über ein Icon löschen, bzw. über ein gut sichtbares Schaltelement eine neue E-Mail-Bestätigung anlegen.

Der Nachteil dieses Ansatzes deutet sich jedoch bereits in Abbildung 7.15 an. Sobald ein Benutzer verschiedene Kriterien oder Versandzeitpunkte für seine E-Mail-Berichte anlegt, verliert diese Ansicht erheblich an Übersicht und der Benutzer hat ggf. nicht mehr das gesamte Dialogfenster auf dem Bildschirm sichtbar und muss scrollen, um alle Einstellungen einsehen zu können. Insbesondere im Hinblick auf den vierten Schritt, welcher in identischer

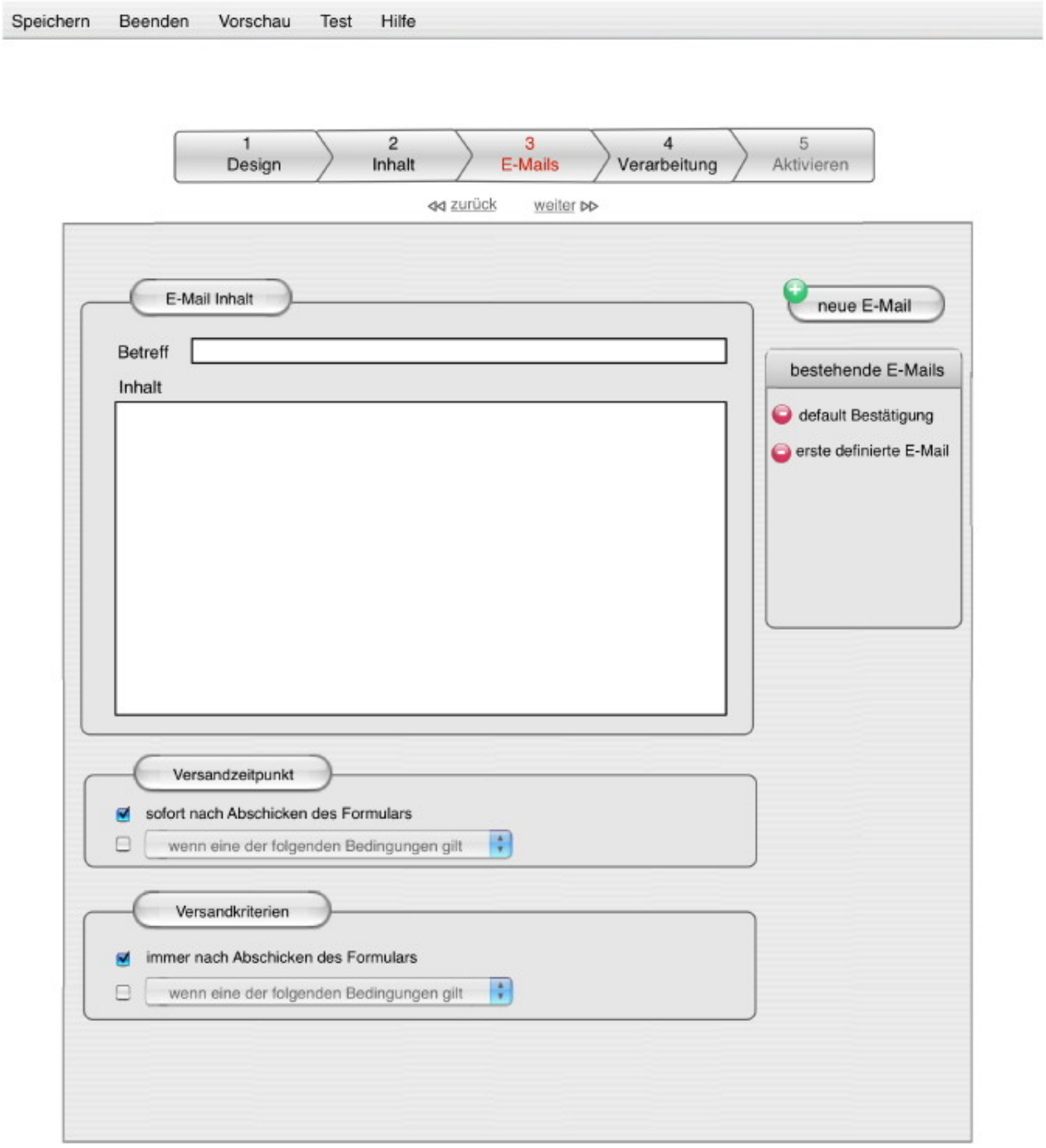


Abbildung 7.14: Default-Ansicht zur Bearbeitung der E-Mail-Bestätigungen in einem Dialogfenster

Weise aufgebaut sein wird wie der dritte Schritt, der aber mehr gegliederte zentrale Funktionen umfasst, ist dieser Ansatz von erheblichem Nachteil.

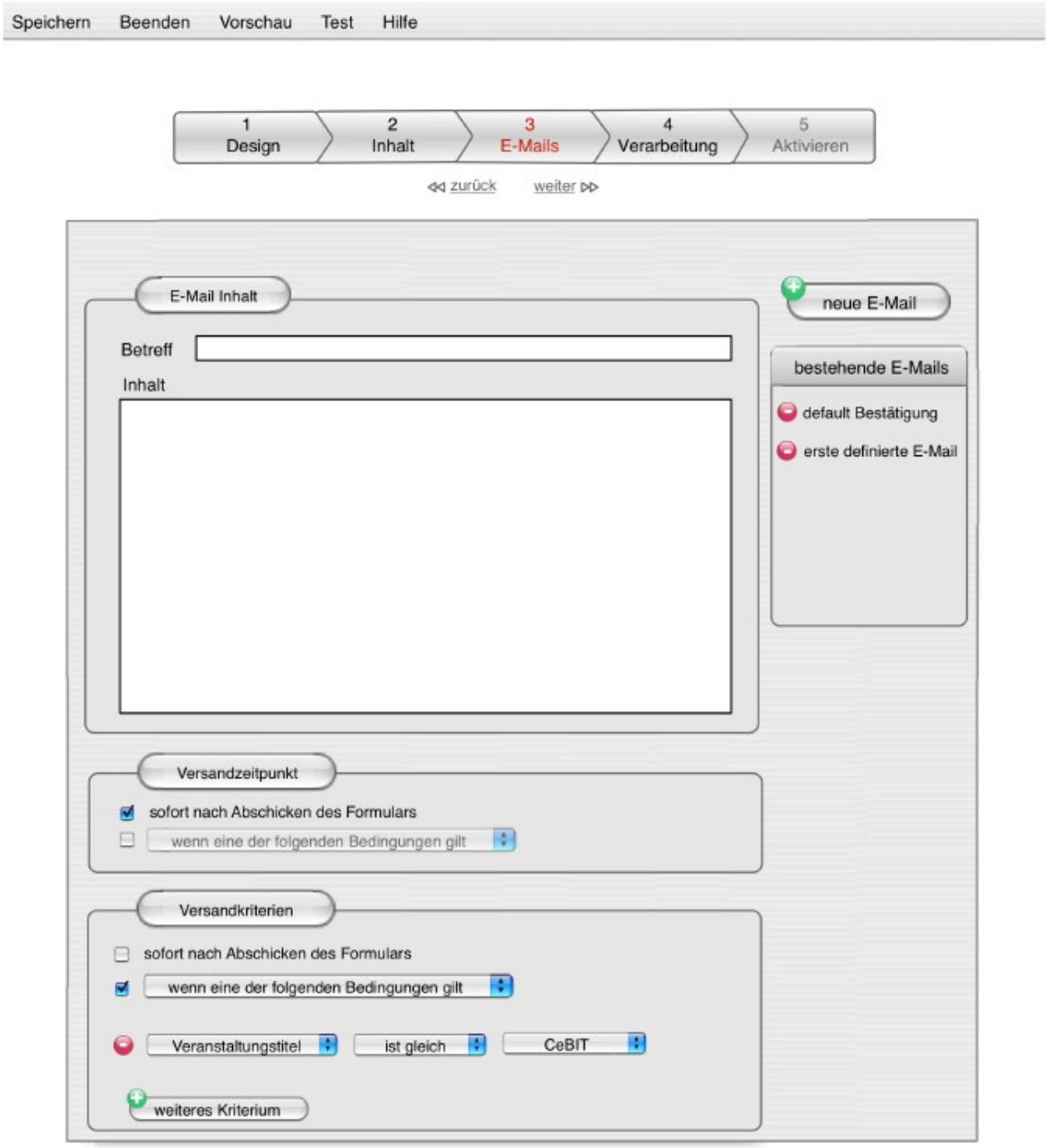


Abbildung 7.15: Erweiterte Ansicht zur Bearbeitung der E-Mail-Bestätigungen in einem Dialogfenster

Eine Verbesserung in diesem Sinne stellt der in Abbildung 7.16 und 7.17 gezeigte Ansatz dar. Hier ist jede zentrale Funktionen durch einen eigenen Dialog repräsentiert. Der Benutzer hat, analog zu den Bedienelementen der Bearbeitungsmodi im zweiten Schritt, jederzeit eine Übersicht über die vorhandenen Funktionen, wird aber nur mit den für ihn momentan relevanten Funktionen konfrontiert und nicht mit den Details der jeweils anderen Funktionen überflutet. Dieser Ansatz der dedizierten Dialogfenster für jede zentrale Funktion ist sehr gut erweiterbar, ohne dass die Übersichtlichkeit des gesamten Schrittes dadurch negativ beeinflusst wird.

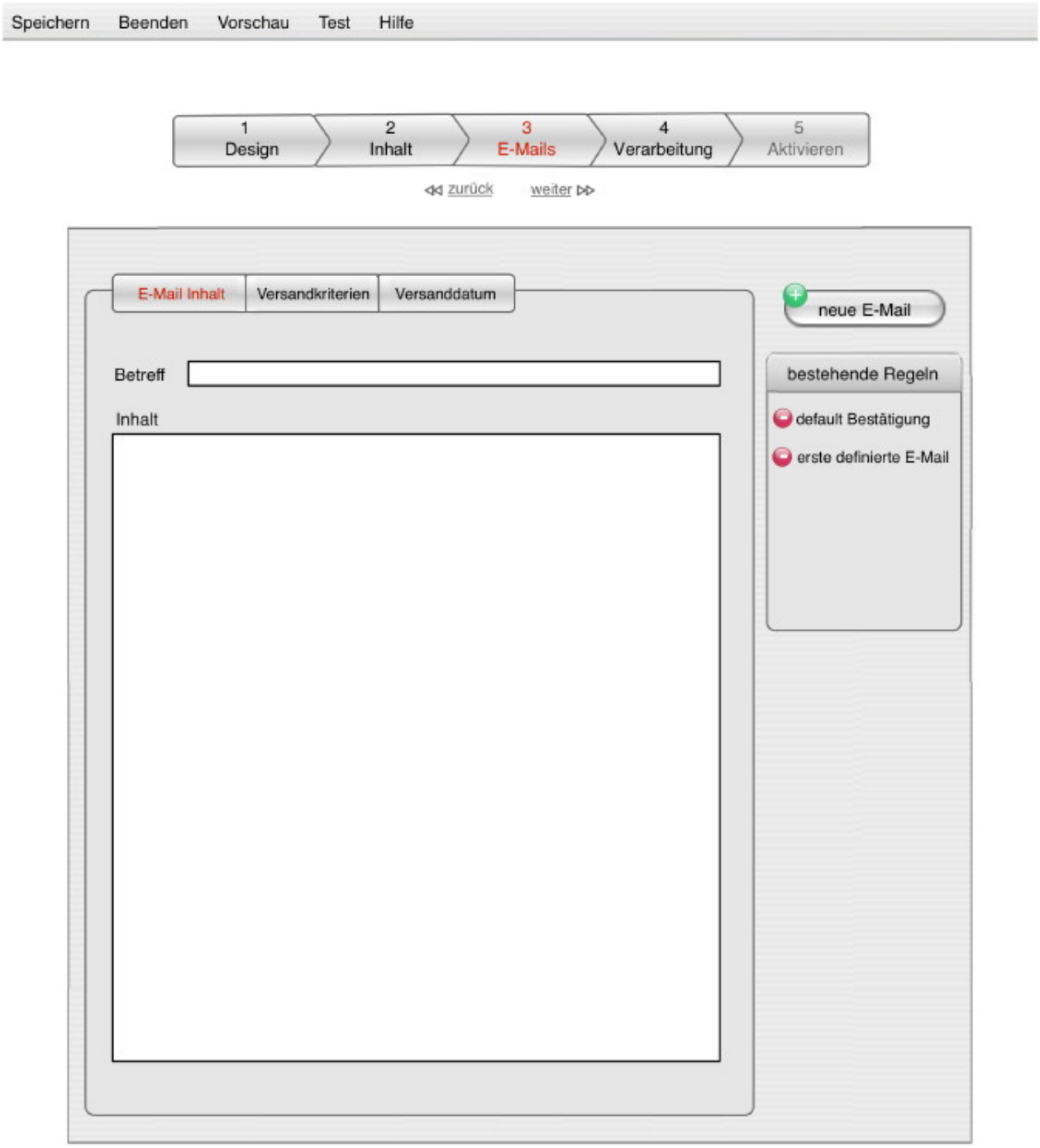


Abbildung 7.16: Bearbeitung des Inhalts der E-Mail-Bestätigungen bei mehreren Dialogfenstern

Abbildung 7.17 verdeutlicht, dass durch diesen Ansatz nicht nur der gesamte Modellierungsschritt an Strukturiertheit gewinnt, sondern durch diese Komplexitätsreduktion auch die einzelnen Funktionen erheblich übersichtlicher sind.

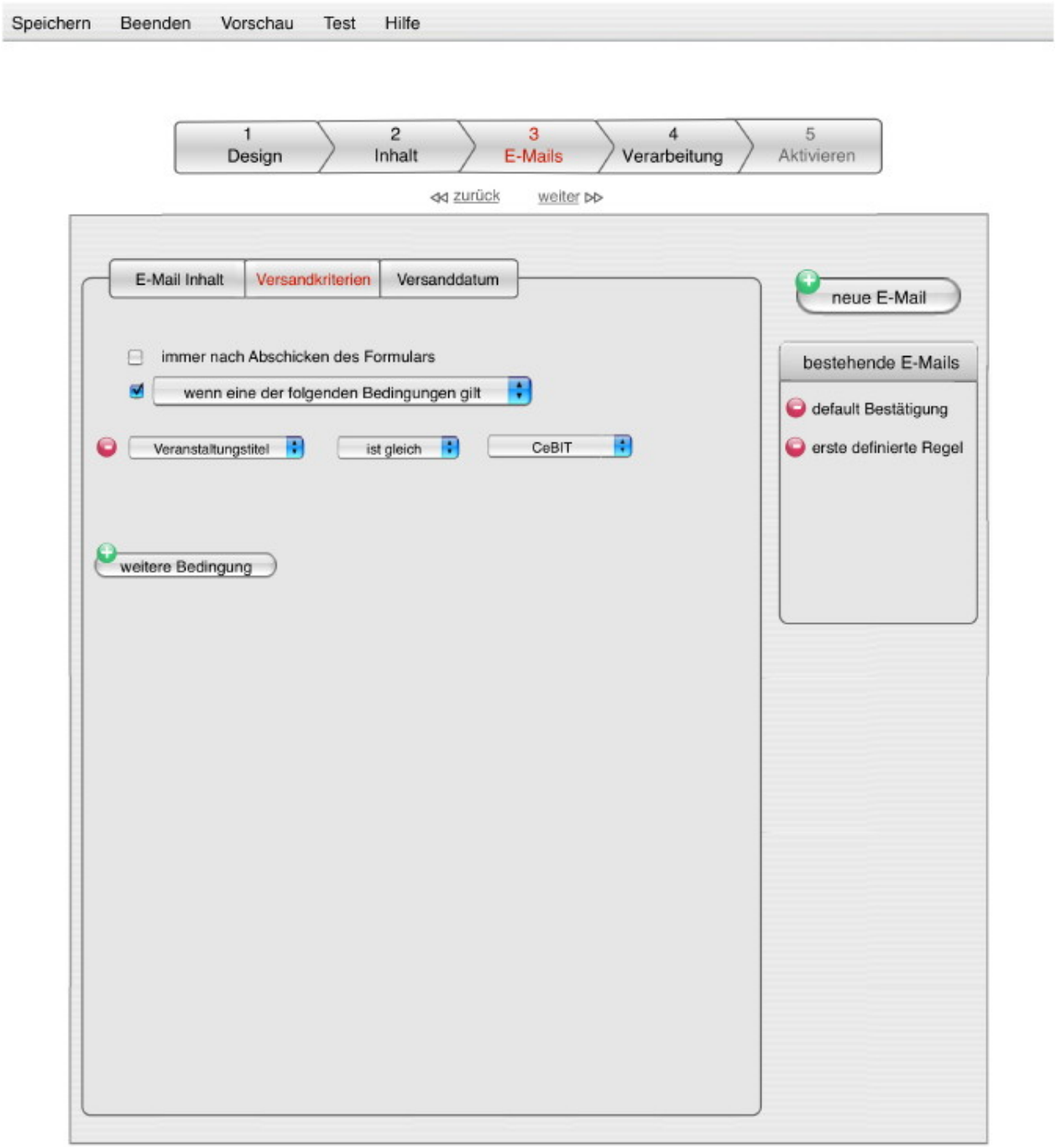


Abbildung 7.17: Bearbeitung der Versandkriterien der E-Mail-Bestätigungen bei mehreren Dialogfenstern

7.4.4 Hochladen des Formulars

Der abschließende Schritt muss dem Benutzer in einem Dialogfenster beispielsweise die URL des Formulars oder eine andere Methode der Weiterverarbeitung anzeigen und gut verständliche Anweisungen geben, wie der Benutzer das Formular in sein Portal einbinden kann.

8 Technische Realisierung

Nachdem in den vergangenen Abschnitten weitreichende Überlegungen bzgl. der Anforderungen und Konzepte der zu entwickelnden Methode vorgenommen wurden, sollen diese Überlegungen technisch umgesetzt werden.

Wie auch die getesteten Softwareprodukte, soll die entwickelte Methode als eine Rich Internet Application realisiert werden. Eine Implementierung als Rich Internet Application wird im Kontext der betrachteten Anwendungsdomäne als sinnvoll erachtet, da es insbesondere im Hinblick auf den Einsatz in Portalumgebungen von Interesse ist, dass keine Installationen bei den Anwendern notwendig sind, sondern dass sie die Anwendung in ihrem Internet Browser ausführen können und die erstellten Daten zentral abgespeichert werden.

Da ein zentraler Aspekt der betrachteten Domäne die visuelle Erstellung von Online-Formularen ist, ist es sinnvoll, diese Formulare auch entsprechend in der technischen Umgebung zu erstellen, in der sie später eingesetzt werden. Dieses bezieht sich darauf, dass in der von den Benutzern gewünschten Formularvorschau schon mit den Stylesheets des Portals gearbeitet werden kann und somit die WYSIWYG-Darstellung des Formulars möglichst nah an dem Endergebnis ist. Auch das von den Benutzern bevorzugte Drag-and-Drop Bedienungskonzept kann durch entsprechende Javascript-Frameworks sehr gut in einer Webanwendung umgesetzt werden.

8.1 Verwendete Technologien

Die Webanwendung zur Realisierung der entwickelten Methode, welche im Folgenden mit dem Namen ***perForm*** bezeichnet wird, wird auf Basis des *Ruby on Rails* Frameworks implementiert werden. Es handelt sich bei diesem Framework um ein quelloffenes Framework für Webanwendungen, welches die Model-View-Controller-Architektur vorgibt und die Prizipien 'Don't Repeat Yourself' (DRY) und 'Convention Over Configuration' verfolgt. Aufgrund der Architektur und der gebotenen Unterstützung des Frameworks sowie den genannten Prinzipien eignet es sich für eine schnelle Umsetzung von Anforderungen und somit ein agiles Vorgehen bei der Entwicklung.

Um eine möglichst flüssige Benutzung der Anwendung *perForm* zu gewährleisten, erfolgt die Kommunikation zwischen Client und Server innerhalb eines Bearbeitungsschrittes asynchron. Jede Einstellung oder Eigenschaft, die ein Benutzer bearbeitet, wird sofort von der Anwendung erfasst und gespeichert, um die Konsistenz beispielsweise des WYSIWYG Formulars oder einer E-Mail-Benachrichtigung auf Clientseite und Serverseite jederzeit zu gewährleisten. Eingaben werden sofort serverseitig validiert, um entweder dem Benutzer in Echtzeit ein Feedback bei Falscheingaben geben zu können oder bei erfolgreicher Speicherung diesen Status sofort auch clientseitig zu aktualisieren. Wichtigster Aspekt ist dabei natürlich, dass für den Benutzer die Kommunikation mit dem Server weitgehend transparent sein muss, das Laden vollständiger Seiten also zu vermeiden ist. Deshalb wird die Kommunikation mit dem Server mit der Ajax (Asynchronous JavaScript and XML) Technologie erfolgen.

Für die Implementierung der Serverkommunikation und der Benutzerinteraktion wird das Prototype JavaScript Framework[1], sowie die darauf aufbauende Bibliothek script.aculo.us[2] eingesetzt. Prototype und script.aculo.us sind bereits in das *Ruby on Rails* Framework integriert und werden von diesem durch so genannte Helper-Methoden sehr gut unterstützt. Helper-Methoden sind in Ruby geschriebene Methoden, welche das *Ruby on Rails* Framework bereit stellt und die dazu dienen sollen, dass der Entwickler beispielsweise HTML- oder JavaScript-Code nicht selbst schreiben muss. Den Methoden werden entsprechende Parameter übergeben und diese generieren daraus den HTML- oder JavaScript-Code.

Das Prototype JavaScript Framework wird für die Verwendung der Ajax Technologie sowie der entsprechenden Document Object Model (DOM) Manipulation zum Einsatz kommen. Die script.aculo.us Bibliothek bietet Umsetzung für die Umsetzung des Drag-and-Drop Bedienungskonzeptes sowie verschiedener visueller Effekte.

Außerdem wird für die automatische Anzeige von Hinweistexten, welche sich kurzzeitig über die Benutzeroberfläche 'legen' sollen, die JavaScript Bibliothek Lightwindow [3] verwendet.

8.2 Architektur

Wie bereits erwähnt, wird durch das *Ruby on Rails* Framework die Model-View-Controller Architektur vorgegeben. Für jedes Model existiert in der Rails Anwendung eine eigene Tabelle und ein entsprechender Controller zur Verwaltung der CRUD-Operationen (Create, Read, Update und Delete) dieses Models sowie Methoden zum Anzeigen aller oder einer Model-Instanz. Per Konvention wird die zu einem Model gehörende Datenbanktabelle in der Mehr-

[1] siehe http://www.prototypejs.org/
[2] siehe http://script.aculo.us/
[3] siehe http://www.stickmanlabs.com/lightwindow/

zahl des Models benannt und der Name des Controllers setzt sich in Camel Case-Schreibweise zusammen aus dem Namen des Models und dem Suffix ‘Controller’.

8.2.1 Models

Als zu persistierende Objekte wurden für die zu entwickelnde Anwendung *perForm* die folgenden Models identifiziert:

Form
Repräsentiert das Formular, jedoch nicht im Sinne eines HTML-Formulars, sondern als das Objekt, welches alle Informationen, die innerhalb des Modellierungsprozesses erstellt werden, verknüpft. Über das Form Objekt werden die Informationen zu den im HTML-Formular enthaltenen Formularelementen, den definierten E-Mail-Benachrichtigungen und dem für das HTML-Formular ausgewählten Design erfasst.

FormElement
Dieses Model dient zur Abspeicherung der Eigenschaften eines einzelnen HTML-Formularelements, wie beispielsweise einem Textfeld und dessen Größe, Validierung, Position, etc. Die Anzahl der für HTML-Formularelemente möglichen Attribute ist relativ begrenzt, da viele Attribute, z.B. die Attribute ‘value’, ‘maxlength’ oder auch ‘size’ in den unterschiedlichen Kontexten der Formularelemente wiederverwendet werden. Daher werden die verschiedenen HTML-Formularelemente nicht als separate Model-Klassen umgesetzt, sondern in Anlehnung an das Single-Table-Inheritance Prinzip alle in einer Tabelle abgespeichert, welche alle für Formularelemente möglichen Attribute enthält. Die Tabelle enthält ein Feld, welches den Typ des Formularelements enthält, also beispielsweise ‘textarea’ oder ‘select’. Auch ist die Benennung der Attribute in der Datenbanktabelle weitgehend identisch mit der Benennung der Attribute in der HTML-Darstellung der Formularelemente, was bei der Programmierung eine erhebliche Vereinfachung bei der HTML-Darstellung eines Formularelements aus der Datenbank bedeutet. Ein Formularelement ist einem Formular zugeordnet und ein Formular kann mehrere Formularelemente enthalten.

ElementCondition Eine ElementCondition stellt eine Bedingung dar, welche in den Anzeigeoptionen eines Formularelements definiert werden kann. Die Bedingung gehört zu einem Formularelement und bezieht sich auf den Wert eines anderen Formularelements. Das Formularelement soll also beispielsweise sichtbar sein, wenn das Formularelement X den Wert y hat.

FormDesign In diesem Model wird ein bestimmtes Layout für Formulare beschrieben, welches dann einem Formular zugewiesen werden kann. Die Definition und Zuweisung eines Layouts zu einem Formular ist jedoch im Rahmen des Prototypen nicht umgesetzt. Einem Formular soll genau ein Layout zugeordnet sein.

EmailNotification Verfasst der Domänenexperte eine E-Mail-Bestätigung, welche an den Benutzer des Formulars verschickt werden soll, so wird diese in dem Model EmailNotification persistiert. Außerdem kann der Domänenexperte Zeitpunkte angeben, zu denen die E-Mail-Bestätigung verschickt werden soll. Außerdem ist es möglich, Versandkriterien anzugeben, unter denen die E-Mail-Bestätigung versendet werden soll. Da für eine E-Mail-Bestätigung mehrere Kriterien definiert werden können, gibt das Attribut 'cond_composition' an, ob nur eine oder alle dieser Kriterien erfüllt sein müssen, damit die E-Mail-Bestätigung verschickt wird. Eine E-Mail-Bestätigung ist einem Formular zugeordnet und ein Formular kann mehrere E-Mail-Benachrichtigungen enthalten.

NotificationCondition Eine NotificationCondition repräsentiert eine Bedingung, die für eine E-Mail-Benachrichtigung gelten muss, damit diese verschickt wird. Sie entspricht also einem Versandkriterium für eine E-Mail-Benachrichtigung und definiert für ein Formularelement, dass dieses einen bestimmten Wert haben muss oder nicht haben darf. Bezieht man sich in der Bedingung auf ein Textfeld, so ist es außerdem möglich zu definieren, dass dieses ein bestimmtes Präfix oder Suffix haben muss. Passwortfelder können nicht in Versandkriterien referenziert werden. Wie im letzten Punkt erwähnt, ist eine NotificationCondition einer E-Mail-Bestätigung zugeordnet und eine E-Mail-Bestätigung kann wiederum mehrere NotificationCondition, also Versandkriterien, haben.

EmailProcessing E-Mails, welche an (portal)interne Benutzer beim Eingang von Formulardaten versendet werden sollen, werden durch das Model EmailProcessing repräsentiert. Sie sollen der Weiterverarbeitung der Formulardaten dienen. Da die (internen) E-Mails zur Verarbeitung der Formulardaten jedoch komplexere Funktionalität unterstützen sollen als normale E-Mail-Bestätigungen an die Benutzer des Formulars, sind diese durch eine eigene Modelklasse vertreten.

ProcessingCondition Analog zu den Versandkriterien für E-Mail-Bestätigungen können auch Versandkriterien für die E-Mail zur Verarbeitung der Formulardaten definiert werden und es gelten zwischen diesen die gleichen Assoziationen wie für E-Mail-Bestätigungen und deren Versandkriterien.

Die Abbildung der 'enthält'- bzw. 'gehört zu'-Assoziationen zwischen Models geschieht im *Ruby on Rails* Framework durch den Aufruf der Methoden `has_many` bzw. `belongs_to` innerhalb der Models. Außerdem muss die Datenbanktabelle des Models auf der `belongs_to` Seite eine Spalte enthalten, die einen Fremdschlüssel auf die `Id` des Models auf der `has_many` Seite darstellt.

Beispielsweise enthält das Model Form den Aufruf der Methode `has_many` für die Models FormElement, EmailNotification, EmailProcessing und den Aufruf der Methode `has_one` für das Model FormDesign

```
class Form < ActiveRecord::Base
  has_many :form_elements
  has_many :email_notifications
  has_many :email_processings
  has_one :form_designs
  ...
end
```

Listing 8.1: Auszug aus dem Form-Model

Abbildung 8.1 zeigt eine Übersicht der Rails Datenbanktabellen für die identifizierten Models. Durch das *Ruby on Rails* Framework werden jeder Datenbanktabelle automatisch die Spalten `id`, `created_at` und `updated_at` hinzu gefügt. Die `id` dient zur eindeutigen Identifizierung jeder Model-Instanz, und in Spalten `created_at` und `updated_at` wird gespeichert, wann die Model-Instanz erstellt, bzw. zuletzt aktualisiert wurde. Diese Spalten sind in der Abbildung 8.1 nicht explizit aufgeführt.

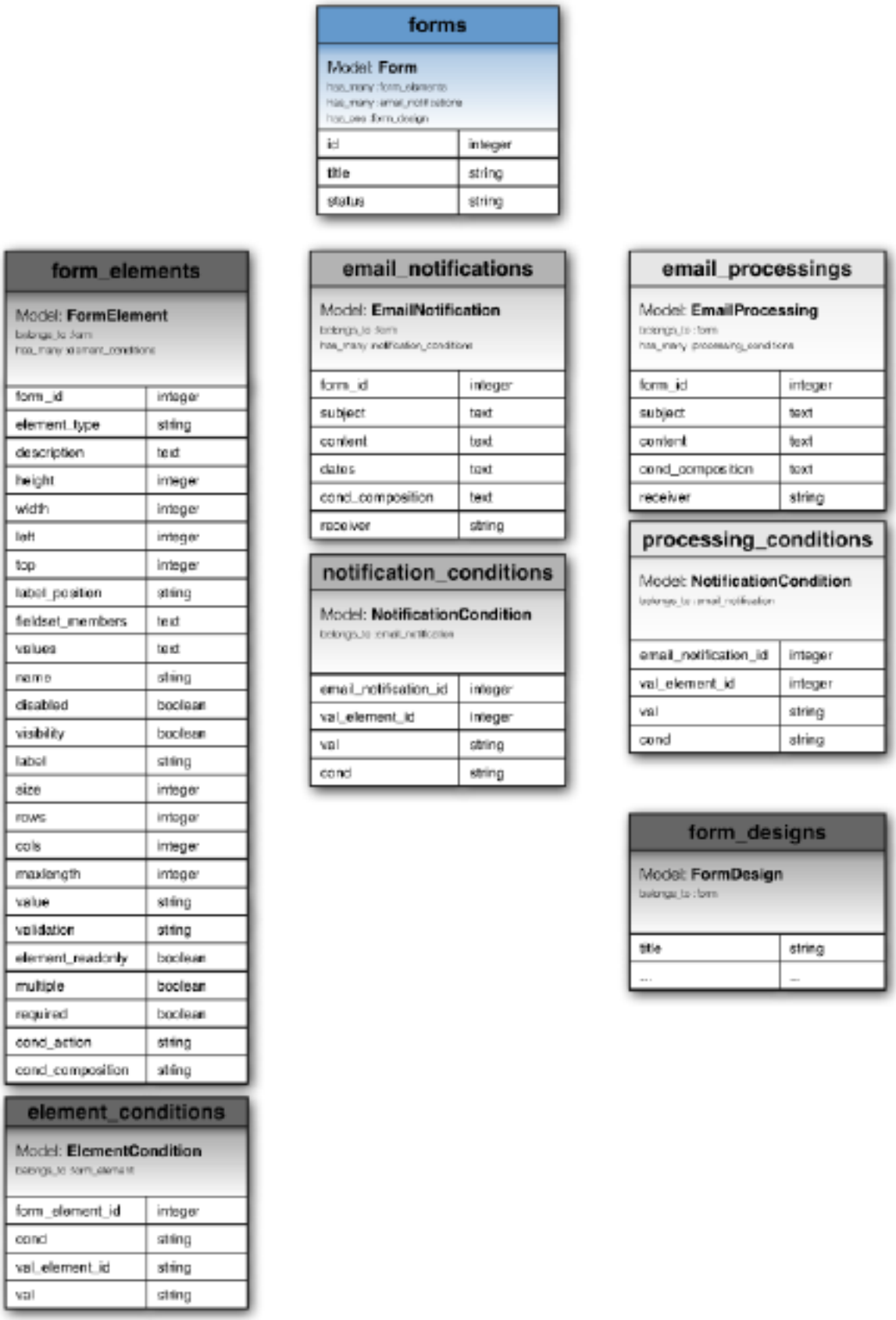


Abbildung 8.1: *perForm*: Datenbanktabellen für die identifizierten Models

Neben den bereits erwähnten Controllern für die CRUD-Operationen der Models gibt es noch einen Controller, der die Informationen über den Status des gesamten Modellierungsprozess hat. Dieser weiß unter anderem, in welchem Bearbeitungsschritt sich der Benutzer gerade befindet. Zusätzlich wird für jeden Schritt des Modellierungsprozesses ein eigener Controller eingeführt. Diese Controller sind jeweils innerhalb eines Verarbeitungsschrittes des Modellierungsprozess dafür verantwortlich, die Ajax-Requests der Clients entgegenzunehmen, die Request-Parameter zu verarbeiten, entsprechende CRUD-Operationen der Models aufzurufen und dann die Response mit den JavaScript-Anweisungen an den Client zurückzuschicken.

Im zweiten Bearbeitungsschritt, also dem Hinzufügen und Bearbeiten von Formularelementen, wird beispielsweise bei einem Drop-Event in der Anwendung, also dem 'Fallenlassen' eines Formularelements über dem WYSIWYG-Formular, per Ajax-Request die `handle_element_drop`-Methode des Controllers aufgerufen, der für die Bearbeitungsvorgänge im zweiten Schritt zuständig ist. Die Methode `handle_element_drop` ist in Listing 8.2.1 gezeigt.

```
class Steps::FormContentController < ApplicationController
  before_filter(:get_form_id)

  def handle_element_drop
    respond_to do |format|
      format.js do
         begin
           FormElement.find params[:id]
         rescue
           @dropped_element = FormElement.create params[:element].merge(
               @form_id)
         else
           @dropped_element = FormElement.update params[:id], params[:element].
               merge(@form_id)
         end
      end
    end
  end

  ...
end
```

Listing 8.2: handle_element_drop-Methode

8.3 Weitere Aspekte der Umsetzung

Wie bereits in Abschnitt 7.4.3 erläutert, soll das Einstellen von Versandkriterien für E-Mails nach dem Konzept des Definierens von Filterregeln in E-Mail Programmen erfolgen. Um in möglichst vielen Punkten bekannte Konzepte einzusetzen, wird das Einstellen von Anzeigeoptionen für Formularelelemte konzeptionell analog erfolgen. Dabei soll weitgehend von manuellen Einträgen der Benutzer abgesehen werden, um Falscheingaben zu vermeiden.

Möchte ein Benutzer beispielsweise für ein Formularelement eine Abhängigkeit zu einem anderen Formularelement definieren, so werden ihm zuerst in einer Auswahlliste die Titel aller weiteren im Formular vorhandenen Formularelemente angezeigt. Zur Veranschaulichung werden hier vorausgreifend Abbildungen der Anwendung *perForm* gezeigt. Abbildung 8.2 zeigt die Auswahl eines Formularelements bei der Definition von Anzeigekriterien.

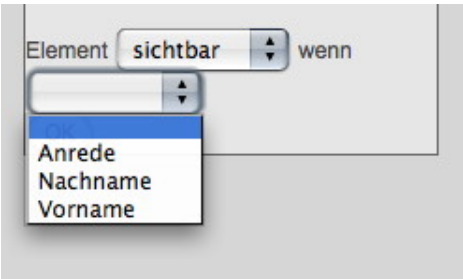


Abbildung 8.2: *perForm*: Auswahl eines Formularelements bei der Definition eines Anzeigekriteriums

Das Formularlement für das die Abhängigkeit definiert werden soll sowie Formularelemente vom Typ Passwortfeld (Abfrage des Attributs `element_type` der Datenbanktabelle `form_elements`) werden in dieser Liste nicht aufgeführt. Existiert für das Formular nur ein einziges Formularelement, so kann keine Abhängigkeit zwischen Formularelementen hergestellt werden und der Dialog zum Definieren von Anzeigekriterien wird ausgeblendet. Die `id` des ausgewählten Formularelements wird in dem Attribut `val_element_id` in der Datenbanktabelle `element_conditions` gespeichert.

Als nächstes muss der Benutzer die Bedingung auswählen, die für den Wert, also die vom Formular-Benutzer eingegebenen Daten, in dem ausgewählten Formularelement gelten soll. Diese Bedingung wird in dem Attribut `cond` der Datenbanktabelle `element_conditions` gespeichert. In Abhängigkeit von dem Typ des ausgewählten Formularelements kann der Benutzer unterschiedliche Bedingungen auswählen. Abbildung 8.3 zeigt die zur Auswahl stehenden Bedingungen für Formularelemente, bei denen der Formular-Benutzer vordefinierte Werte auswählen kann, wie beispielsweise bei Radiobuttons, Checkboxen oder Auswahllisten.

Abbildung 8.3: *perForm*: Mögliche Bedingungen für Formularelemente mit fest definierten Werten

Ist das ausgewählte Formularelement ein Eingabefeld, bei dem der Formular-Benutzer Texteingaben machen kann, so stehen andere Bedingungen bei der Definition des Anzeigekriteriums zur Verfügung. Dies wird in Abbildung 8.4 gezeigt.

Abbildung 8.4: *perForm*: Mögliche Bedingungen für Formularelemente bei denen Texteingaben möglich sind

Im nächsten Schritt zur Definition eines Anzeigekriteriums muss der Benutzer den Wert angeben, der für die Bedingung gelten soll. Dieser Wert wird im Attribut `val` der Datenbanktabelle `element_conditions` gespeichert. Handelt es sich bei dem für das Anzeigekriterium ausgewählten Formularelement um ein Element mit vordefinierten möglichen Werten, so werden diese dem Benutzer wie in Abbildung 8.5 illustriert, zur Auswahl angeboten.

Ist das ausgewählte Formularelement jedoch ein Feld für Texteingaben des Formular-Benutzers, so kann der Wert für die Bedingung manuell eingegeben werden, was in Abbildung 8.6 illustriert ist.

Nach dem gleichen Konzept wird beim Einfügen von Formularvariablen in die E-Mail Inhalte und beim Definieren der Versandkriterien für E-Mails vorgegangen. Abbildung 8.7 zeigt dies für die Benachrichtigungskriterien der E-Mails zur Verarbeitung der Formulardaten.

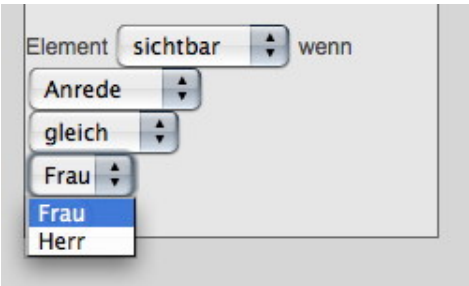


Abbildung 8.5: *perForm*: Werte für Formularelemente mit fest definierten Werten

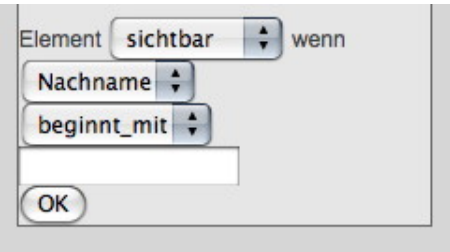


Abbildung 8.6: *perForm*: Werte für Bedingungen von Anzeigekriterien bei denen Texteingaben möglich sind

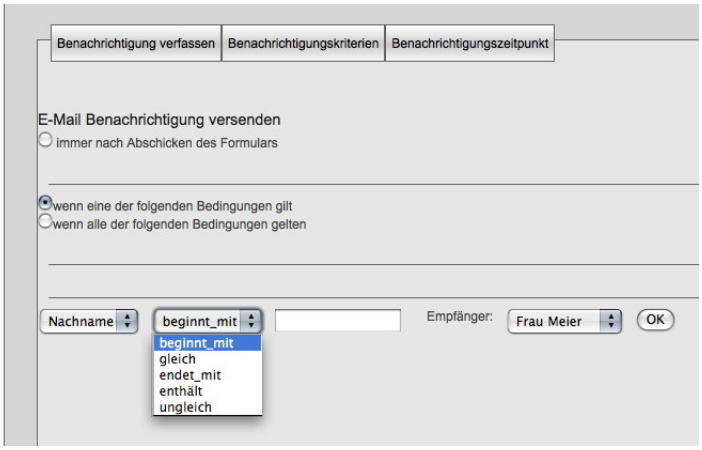


Abbildung 8.7: *perForm*: Definition von Benachrichtigungskriterien

8.4 Qualitätssicherung

Zur Sicherstellung einer hohen technischen Qualität der Anwendung *perForm* wurde diese testgetrieben entwickelt. Es wurde agil vorgegangen und die Methode des Test-Driven Development (TDD) eingesetzt. Konkret wurde das Konzept des Behaviour-Driven Development (BDD) eingesetzt, welches gegenüber dem klassischen TDD Ansatz eine Erweiterung dar-

stellt.

Beim BDD geht es darum, in den Tests das erwartete *Verhalten* der Software zu formulieren. Wie der Autor Dave Astels in [Ast03] beschreibt, basiert die Entwicklung des BDD Ansatzes darauf, dass Tests mit der Zeit dazu führen, dass Entwickler "ihr API" entdecken und vor der Implementierung in den Tests spezifizieren. Der Autor betont, dass es jedoch beim TDD nicht um das *Testen* geht, sondern vielmehr darum, das *Verhalten* der Software zu definieren. Aus diesem Gedanken heraus entwickelte der Autor eine Bibliothek, um Verhalten (Behaviour) zu formulieren.

Da dieses Konzept sehr ausdrucksstark ist, wurde dieses für Ruby in die Bibliothek *rspec*[4] aufgenommen und weiterentwickelt. *RSpec* ist eine von *Ruby on Rails* unabhängige Bibliothek, mittels der man definieren kann, welche Anforderungen ein Programm erfüllen muss. Installiert man *RSpec* für eine *Ruby on Rails* Anwendung, so wird innerhalb der Anwendung ein Top-Level Ordner mit dem Namen spec angelegt. Innerhalb des Ordners befinden sich Verzeichnisse für die controller, models, views, fixtures, helper und das routing, in denen sich dann die Dateien mit den entsprechenden Tests befinden. Abbildung 8.8 zeigt das spec-Verzeichnis im Kontext des Verzeichnisbaums der gesamten Anwendung. Zu sehen sind dort auch die Dateien, in denen die Tests für die Controller definiert sind.

[4] siehe http://rspec.info/

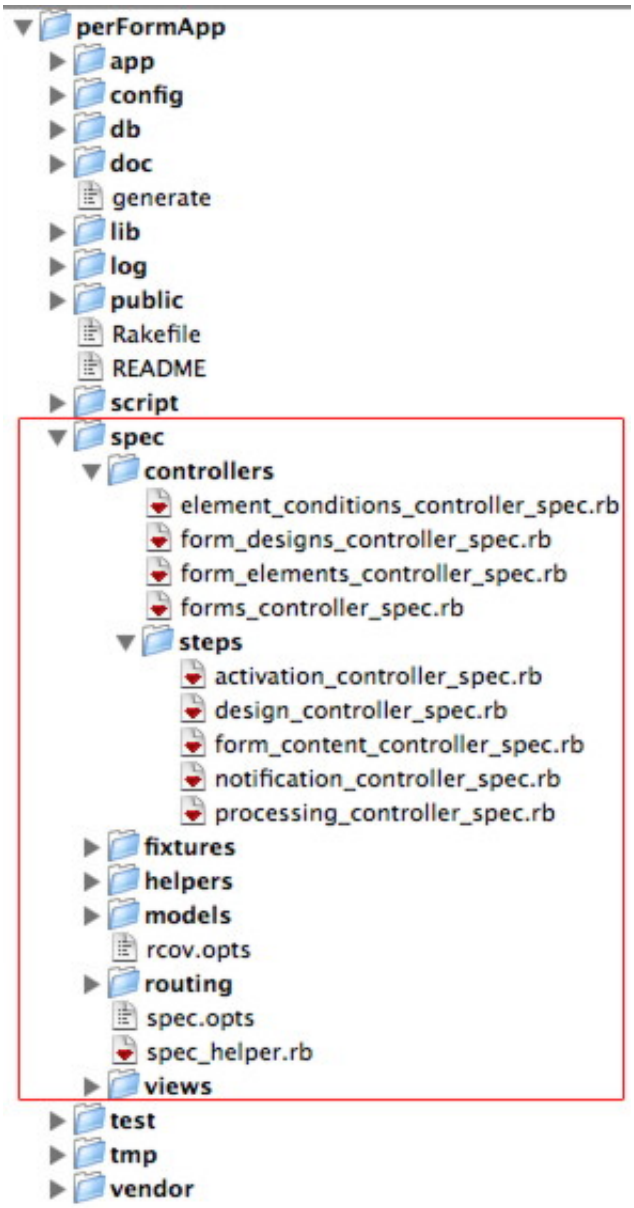


Abbildung 8.8: Das spec-Verzeichnis in Kontext einer *Ruby on Rails* Anwendung

Das Verhalten der Software wird in *RSpec* in 'describe'-Blöcken beschrieben, wobei innerhalb eines solchen describe-Blocks die verschiedenen Fälle (Examples) für dieses Verhalten definiert werden. Abbildung 8.9 zeigt einen Ausschnitt aus dem in *RSpec* formulierten Verhalten des Controllers, der für die Verarbeitung der Ajax-Requests für das Hinzufügen und Bearbeiten von Formularelementen zuständig ist.

```
describe Steps::FormContentController,  "handle a dropped element" do
  def do_xhr_text_id
    xhr :post, :handle_element_drop, :id => "draggable_textarea",
    :element => {"element_type"=>"draggable_textarea", "top"=>"50", "height"=>"42",
      "label"=>"draggable_textarea", "left"=>"-250", "width"=>"220"}
  end

  it "should have a route for this controller and action" do
    route_for(:controller => "steps/form_content",
    :action => "handle_element_drop").should == "/steps/form_content/handle_element_drop"
  end

  it "should respond to the action for handling a dropped element" do
    controller.should respond_to(:handle_element_drop)
  end

  it "should create a new Element with the correct Properties if the dropped Element
        does not exist in the database" do
    FormElement.should_receive(:find).and_raise(:RecordNotFound)
    FormElement.should_receive(:create)

    do_xhr_text_id
  end

  it "should insert a new Form Element in the database if the Element does not exist" do
    FormElement.should_receive(:find).and_raise(:RecordNotFound)

     lambda {
       do_xhr_text_id
     }.should change(FormElement, :count).by(1)
  end
```

Abbildung 8.9: Das in *RSpec* formulierte Verhalten für die Controller-Methode, die bei einem Drop-Event aufgerufen wird

Ist das durch die Tests spezifizierte Verhalten der Anwendung erfüllt, so sieht eine Ausgabe der Testergebnisse beispielsweise wie in Abbildung 8.10 aus.

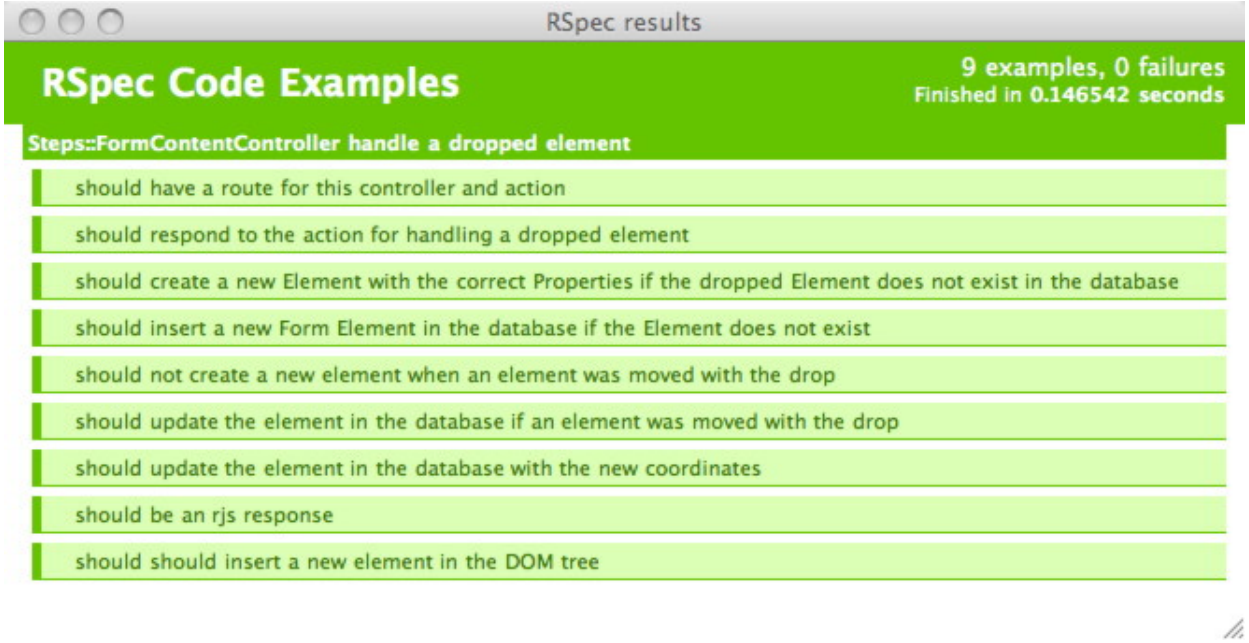


Abbildung 8.10: Ausgabe der *RSpec* Tests bei Erfüllung des spezifizierten Verhaltens

Anmerkung:

Die im Rahmen der Entwicklung der Anwendung *perForm* geschriebenen Tests sind Tests, welche auf den verschiedenen horizontalen Ebenen der Anwendung *perForm* vorgenommen wurden. Automatisierte Integrations- oder Systemtests, welche einem vertikalen Test durch die verschiedenen Ebenen der Anwendung entsprechen, wurden noch nicht implementiert. Dieses ist bei einer Weiterentwicklung der Anwendung unbedingt vorzunehmen, konnte jedoch im Rahmen dieser Arbeit nicht mehr ausführlich behandelt werden.

9 Verifikation der Überlegungen

Bisher wurde in Abschnitt 3 bestehende Softwaresysteme für die Domäne der Prozesse in Webportalen betrachtet. Anhand einer Auswahl von Repräsentanten wurde deren Gebrauchstauglichkeit für die genannte Domäne durch Usability-Tests in Abschnitt 4 evaluiert. Die erhobenen Testdaten wurden in Abschnitt 5 ausgewertet und analysiert und anhand dessen wurden die Problematiken und Vorzüge dieser Softwaresysteme herausgestellt.

Unter Berücksichtigung dieser Ergebnisse wurde in Abschnitt 6 eine als Produktvision bezeichnete Methode mit bestimmten Anforderungen und zu berücksichtigenden Aspekten entwickelt. Da die Usability dieser Methode im Fokus dieser Arbeit steht, wurde in Abschnitt 7 das konzeptuelle Modell des zu realisierenden Benutzerinterfaces unter Betrachtung verschiedenster Usability-Aspekte diskutiert und entwickelt. Anschließend wurde in Abschnitt 8 das technische Konzept der Anwendung *perForm* entworfen, welche eine Implementierung der entwickelten Methode darstellt.

In diesem Abschnitt soll nun die Verifikation der entwickelten Methode und deren Konzepte anhand von Usabilty-Tests erfolgen. Dafür wird den Probanden ein hinreichend vollständiger Prototyp der Anwendung *perForm* vorgesetzt.

Anzumerken ist, dass aufgrund des Fokus' dieser Arbeit die funktionale Vollständigkeit der Anwendung nur von untergeordneter Bedeutung ist. Denn aufgrund der ausführlichen theoretischen Betrachtungen und Untersuchungen ist es im Rahmen dieser Arbeit nicht möglich eine vollständig ausgereifte Implementierung der entwickelten Methode zu liefern. Vielmehr ging es bei der Umsetzung der Anwendung *perForm* darum, dass die Implementierung die technische Realisierbarkeit der entwickelten Methode aufzeigen soll. Die Konzepte der theoretischen Überlegungen müssen in dem Prototypen also insofern umgesetzt sein, dass diese anhand der Usability-Tests verifiziert werden können.

Auch wenn der Prototyp technisch gesehen funktional nicht vollständig ist, so muss er doch zumindest in der Benutzeroberfläche nahezu die vollständige Funktionalität präsentieren, damit es nicht durch deutlich geringere Komplexität in der Benutzeroberfläche zu Verfälschungen der Testergebnisse gegenüber den evaluierten, bestehenden Softwaresysteme kommt.

9.1 Durchführung von Usability-Tests mit der entwickelten Methode

Die Durchführung der Usability-Tests für den Prototypen erfolgte analog zu der in Abschnitt 4. Sie waren identisch bzgl. der Test-Infrastruktur und der räumlichen Gegebenheiten.

Die Aufgabenstellung lautete wie folgt:

Erstellen Sie ein Online-Formular, mit dem ein Kunde Probleme mit seinem Informationssystem bei Ihnen melden kann.

- Das Formular enthält beschriftete einzeilige Textfelder zur Eingabe des Namens und der E-Mail-Adresse. Beide Felder sind Pflichtfelder und sind als solche gekennzeichnet.
- Das Eingabefeld für die E-Mail-Adresse akzeptiert nur E-Mail-Adressen in einem gültigen Format.
- Der Kunde muss für sein Problem einen Problemgrad vergeben.
- Setzt der Kunde den Problemgrad seiner Problemmeldung auf ‘Hoch’, erscheint eine Auswahlbox, in der er Details zu seinem Problem angeben muss.
- Ist der Problemgrad auf ‘Niedrig’, so ist die Auswahlbox für die Details nicht zu sehen.
- Das Formular enthält ein beschriftetes mehrzeiliges Texteingabefeld, in dem der Kunde seine Problembeschreibung eingeben muss. Das mehrzeilige Textfeld für die Problembeschreibung ist ein Pflichtfeld und als solches gekennzeichnet.
- Der Absender des Formulars bekommt nach dem Abschicken des Formulars eine E-Mail-Benachrichtigung, in der er mit Anrede und Namen begrüßt wird.
- In Abhängigkeit von dem eingegebenen Nachnamen wird das Formular an unterschiedliche Empfänger in Ihrem Unternehmen verschickt.

Daraus ergab sich die folgende Liste der während der Testdurchführung durchzugehenden Tasks:

1 Verschaffen Sie sich einen Überblick über die Anwendung und deren Funktionen.

2 Erstellen Sie ein neues Formular mit dem default-Layout.

3 Fügen Sie dem Formular ein beschriftetes einzeiliges Texteingabefeld zur Eingabe des Nachnamens hinzu. Das Feld zur Eingabe des Nachnamens ist ein Pflichtfeld. Das Textfeld hat eine Breite von 25 Zeichen und es sind auch maximal 25 Zeichen als Eingabe erlaubt.

4 Fügen Sie dem Formular ein beschriftetes einzeiliges Texteingabefeld zur Eingabe der E-Mail-Adresse hinzu. Das Feld zur Eingabe der E-Mail-Adresse ist ein Pflichtfeld. Das Textfeld hat eine Breite von 25 Zeichen und es sind auch maximal 25 Zeichen als Eingabe erlaubt. Das Eingabefeld für die E-Mail-Adresse überprüft E-Mail-Adressen auf ein gültiges Format.

5 Fügen Sie dem Formular ein mit ‘Problemgrad’ beschriftetes Feld hinzu, mit dem der Benutzer zwischen den Optionen ‘Hoch’ oder ‘Niedrig’ wählen kann. Die Option ‘Niedrig’ soll die Vorauswahl sein.

6 Fügen Sie dem Formular ein mit ‘Details’ beschriftete Liste hinzu, bei dem der Benutzer zwischen den Werten ‘Hardware’ und ‘Software’ auswählen kann. Die Liste soll die Höhe 1 haben. Es soll nur einer der beiden Werte auswählbar sein. Legen Sie fest, dass die Auswahl für die Details nur dann zu sehen ist, wenn der Problemgrad ‘Hoch’ ausgewählt ist.

7 Fügen Sie dem Formular ein mit ‘Problem’ beschriftetes mehrzeiliges Texteingabefeld zur Eingabe der Problembeschreibung. Die Problembeschreibung ist kein Pflichtfeld. Das Texteingabefeld hat eine Breite von 25 Zeichen und eine Höhe von 10 Zeichen. Das Texteingabefeld ist mit dem Text ‘Problembeschreibung’ vorbelegt.

8 Legen Sie fest, dass der Absender des Formulars nach dem Abschicken des Formulars eine E-Mail-Bestätigung bekommt, in der er mit seinem Nachnamen begrüßt wird und die den Text ‘Wir haben ihre Problemmeldung erhalten’ enthält. Die Bestätigungs-E-Mail hat den Betreff ‘Vielen Dank’.

9 Legen Sie fest, dass nachdem ein Formular ausgefüllt wurde, eine E-Mail-Benachrichtigung zur Verarbeitung der Formulardaten verschickt wird. Der Betreff der E-Mail-Benachrichtigung soll den Text ‘Ein Formular wurde abgeschickt’ enthalten. Der Inhalt der Benachrichtigungs-E-Mail kann einen beliebigen Text enthalten. Die Benachrichtigungs-E-Mail soll an Herrn Schmidt aus Ihrem Unternehmen gehen, wenn der von dem Benutzer des Formulars eingegebene Nachname mit ‘Sch’ beginnt.

10 Analyse der durchgeführten Usability-Tests mit der entwickelten Methode

Die Analyse der erhobenen Testdaten erfolgte anhand derselben Kriterien wie in Abschnitt 5 und wurde analog durchgeführt. Um auch an dieser Stelle wieder eine differenzierte Analyse vornehmen zu können, wurden die durchgeführten Tasks in Mikrotasks aufgegliedert. Ein Übersicht der Mikrotasks ist in Tabelle 10.1 zu finden.

Die Bewertung der Testdaten erfolgte ebenfalls anhand der in 5.1 erhobenen Metriken. Die ausführliche Tabelle der Auswertungsdaten ist im Anhang E zu finden.

Task	Mikrotask	Beschreibung
1	1	Überblick verschaffen
2	2	neues Formular mit default-Layout erstellen
3	3	Texteingabefeld für den Nachnamen hinzufügen
	4	Texteingabefeld für den Nachnamen beschriften
	5	Texteingabefeld für den Nachnamen als Pflichtfeld setzen
	6	für das Texteingabefeld für den Nachnamen eine Breite von 25 Zeichen einstellen
	7	für das Texteingabefeld für den Nachnamen eine maximale Anzahl Zeichen (25 Zeichen) einstellen
4	8	Texteingabefeld für die E-Mail-Adresse hinzufügen
	9	Texteingabefeld für die E-Mail-Adresse beschriften
	10	für das Texteingabefeld für die E-Mail-Adresse eine Breite von 25 Zeichen einstellen
	11	für das Texteingabefeld für die E-Mail-Adresse eine maximale Anzahl Zeichen (25 Zeichen) einstellen
	12	Texteingabefeld für die E-Mail-Adresse als Pflichtfeld setzen
	13	für das Texteingabefeld für die E-Mail-Adresse die Validierung auf eine gültige Eingabe einstellen
5	14	Optionsfeld für den Problemgrad hinzufügen
	15	Optionsfeld für den Problemgrad beschriften
	16	dem Optionsfeld für den Problemgrad die Optionen ‘hoch’ und ‘niedrig’ hinzufügen
	17	bei dem Optionsfeld für den Problemgrad die Option ‘niedrig’ als Vorauswahl einstellen
6	18	Auswahlliste für die Details hinzufügen
	19	Auswahlliste für die Details beschriften
	20	der Auswahlliste für die Details die Werte ‘Hardware’ und ‘Software’ hinzufügen
	21	bei der Auswahlliste für die Details die Höhe 1 einstellen
	22	Einstellen, dass die Auswahlliste für die Details nur dann zu sehen ist, wenn der Problemgrad ‘hoch’ ausgewählt ist.
7	23	Textbereich zur Eingabe der Problembeschreibung hinzufügen
	24	Textbereich zur Eingabe der Problembeschreibung beschriften
	25	bei dem Texteingabefeld zur Eingabe der Problembeschreibung die Höhe und Breite festlegen
	26	bei dem Texteingabefeld zur Eingabe der Problembeschreibung die Vorbelegung festlegen
8	27	E-Mail-Bestätigung an den Absender des Formulars einstellen
	28	Betreff der E-Mail-Bestätigung an den Absender des Formulars festlegen
	29	Festlegen, dass der Empfänger der E-Mail-Bestätigung mit seinem im Formular eingegebenen Nachnamen begrüßt wird.
	30	Text der E-Mail-Bestätigung an den Absender des Formulars festlegen
9	31	E-Mail zur Verarbeitung der Formulardaten einstellen
	32	Betreff der E-Mail zur Verarbeitung der Formulardaten festlegen
	33	Versandkriterium für die E-Mail zur Verarbeitung einstellen

Tabelle 10.1: *perForm*: Mikrotasks

10.1 Mittelwert der aufgetretenen Usability-Probleme pro Mikrotask

Es wird wie auch in Abschnitt 5.2 zunächst pro Mikrotask aus den Problemgraden beider Usability-Aspekte und aller Probanden der Mittelwert gebildet. Dadurch erhält man eine Übersicht der hinsichtlich aufgetretener Usability-Probleme kritischen Mikrotasks.

Insgesamt kam es bei der Durchführung der Usability-Tests mit der entwickelten Methode zu sehr wenigen Problemen. Nur in Einzelfällen traten schwerwiegende Probleme auf, bei denen ein Eingreifen des Testadministrators notwendig war. Dadurch kann es sein, dass auffällige Abweichungen in den Mittelwerten bereits durch singuläre Schwierigkeiten hervorgerufen sind.

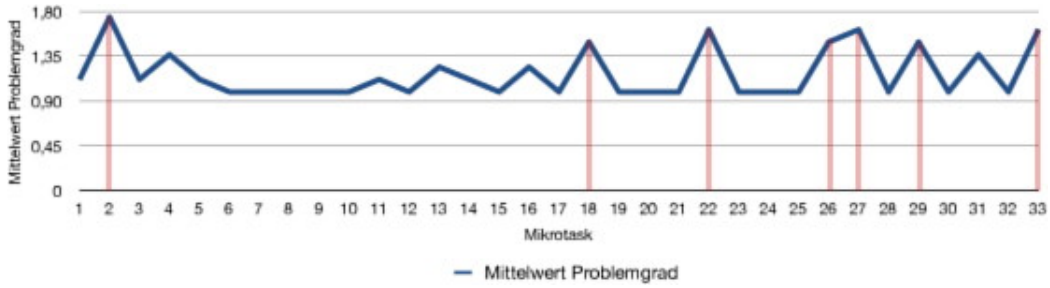


Abbildung 10.1: *perForm*: Mittelwert der aufgetretenen Usability-Probleme aller Probanden pro Mikrotask

Anhand von Abbildung 10.1 sind die folgenden Mikrotasks hinsichtlich ihrer Usability-Probleme als kritisch zu bewerten:

2 ein neues Formular mit dem default-Layout erstellen

18 die Auswahlliste für die Details hinzufügen

22 Einstellen, dass die Auswahlliste für die Details nur dann zu sehen ist, wenn der Problemgrad ‘hoch’ ausgewählt ist.

26 beim Texteingabefeld zur Eingabe der Problembeschreibung die Vorbelegung festlegen

27 E-Mail-Bestätigung an den Absender des Formulars einstellen

29 Festlegen, dass der Empfänger der E-Mail-Bestätigung mit seinem im Formular eingegebenen Nachnamen begrüßt wird.

33 Versandkriterium für die E-Mail zur Verarbeitung einstellen

Abbildung 10.1 zeigt, dass es am Anfang bei der Auswahl des Layouts zu Problemen kam. Im Unterschied zu den evaluierten Softwarelösungen in Abschnitt 5 kam es jedoch bei der Durchführung des jeweils ersten Mikrotasks der die Aspekte 'Hinzufügen' oder 'Bearbeiten von Elementen' betrifft, nicht zu Problemen. Entsprechend ist auch in Abbildung 10.1 keine Lernkurve bei den Mikrotasks dieser Bedienungsaspekte vorhanden. Dies ist darauf zurück zu führen, dass beim 'Betreten' des zweiten Modellierungsschrittes, in dem die Formularelemente hinzugefügt und bearbeitet werden, ein kurzer Hinweis bzgl. der zentralen Bedienungskonzepte angezeigt wird. Die Anzeige dieses Hinweises ist in Abbildung 10.2 zu sehen. Abbildung 10.3 zeigt die Anwendung *perForm* im zweiten Modellierungsschritt im Bearbeitungsmodus für Formularelemente.

Abbildung 10.2: *perForm*: Hinweis zu den Bedienungskonzepten im zweiten Modellierungsschritt

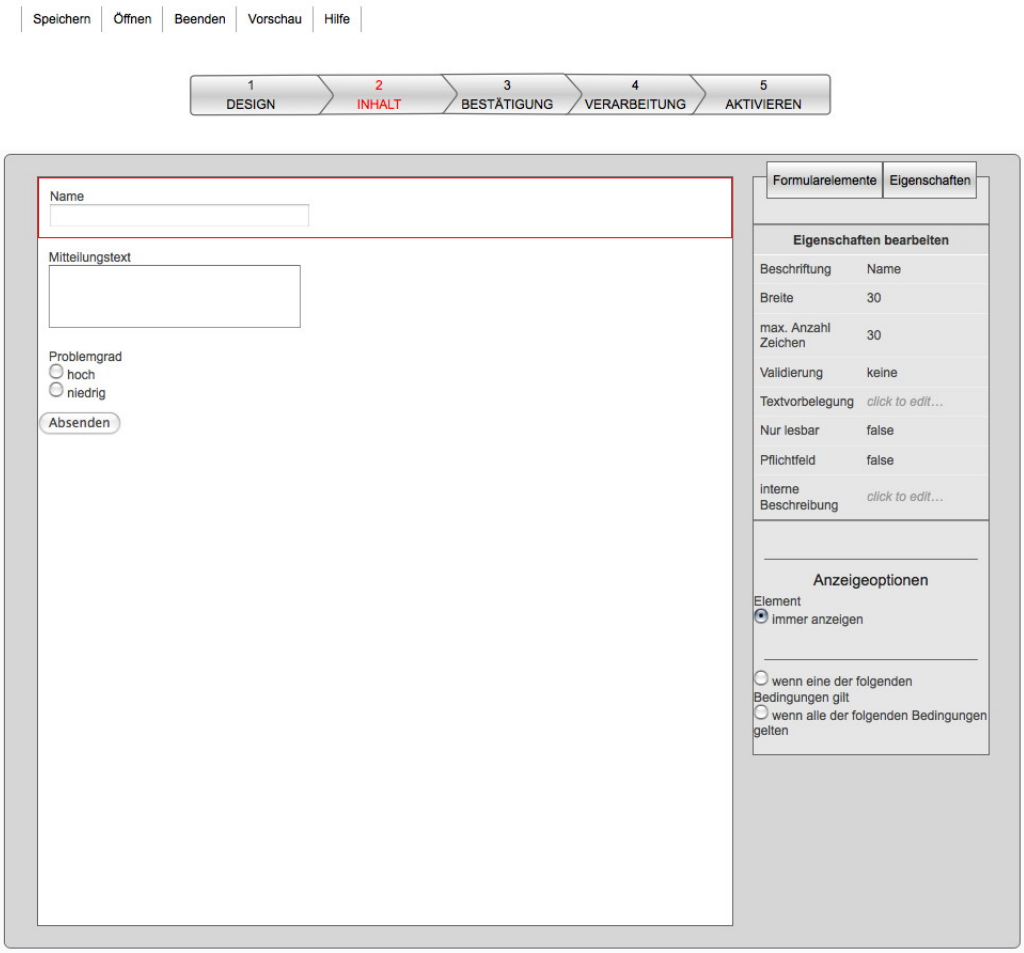


Abbildung 10.3: *perForm*: Bearbeitung der Eigenschaften von Formularelementen

Drei problematische Mikrotasks, die aufgrund eines schwerwiegenden Einzelfalls in die Kategorie der kritischen Mikrotasks fallen, sind die Mikrotasks 18, 26 und 29. Bei diesen Mikrotasks war ein Eingreifen des Testadministrators notwendig, da der Proband nicht das richtige Bedienelement fand.

Der kritische Mikrotask 22 beinhaltet das Einstellen der Anzeigeoptionen für die Auswahlliste mit den Details und ist der erste Mikrotask, der den Aspekt der Anzeigeoptionen beinhaltet. Es war in keinem Fall ein Eingreifen des Testadministrators notwendig, die Hälfte der Probanden suchten jedoch eine etwas längere Zeit (mittelschweres Problem), bis sie das entsprechende Bedienelement erkannten. Bei der Durchführung des Mikrotasks, also dem Einstellen des Anzeigekriteriums, was in Abbildung 10.4 gezeigt ist, gab es keine weiteren Probleme.

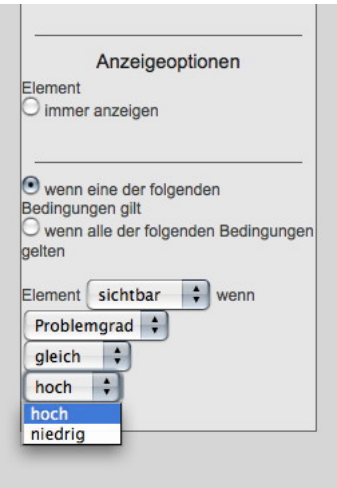


Abbildung 10.4: *perForm*: Einstellen der Anzeigeoptionen für ein Formularelement

Der nächste kritische Mikrotask ist Mikrotask 27, welcher der erste Mikrotask ist, der das Konzept der E-Mail-Bestätigung behandelt. Bei diesem Mikrotask musste der Proband erkennen, dass er den Schritt der Formularerstellung verlassen und in den nächsten Schritt zum Verfassen der E-Mail-Bestätigung wechseln muss. Abbildung 10.5 zeigt den Dialog zum Verfassen der E-Mail-Bestätigung an den Benutzer des Formulars. Die Usability-Probleme, die während der Durchführung dieses Mikrotasks auftraten, waren maximal vom Schweregrad 2, was auf eine kurze Orientierung, aber dann problemlose Durchführung der Probanden an dieser Stelle hinweist.

Der letzte als kritisch zu betrachtende Mikrotask ist Mikrotask 33, der als erster Mikrotask das Einstellen eines Versandkriteriums der E-Mail für die interne Verarbeitung der Formulardaten forderte. Analog zu Mikrotask 22 war in keinem Fall ein Eingreifen des Testadministrators notwendig. Es kam lediglich zu leichten bis mittelschweren Problemen. Abbildung 10.6 zeigt die Ansicht zum Definieren von Kriterien für das Versenden der internen E-Mail-Benachrichtigungen zur Verarbeitung der Formulardaten.

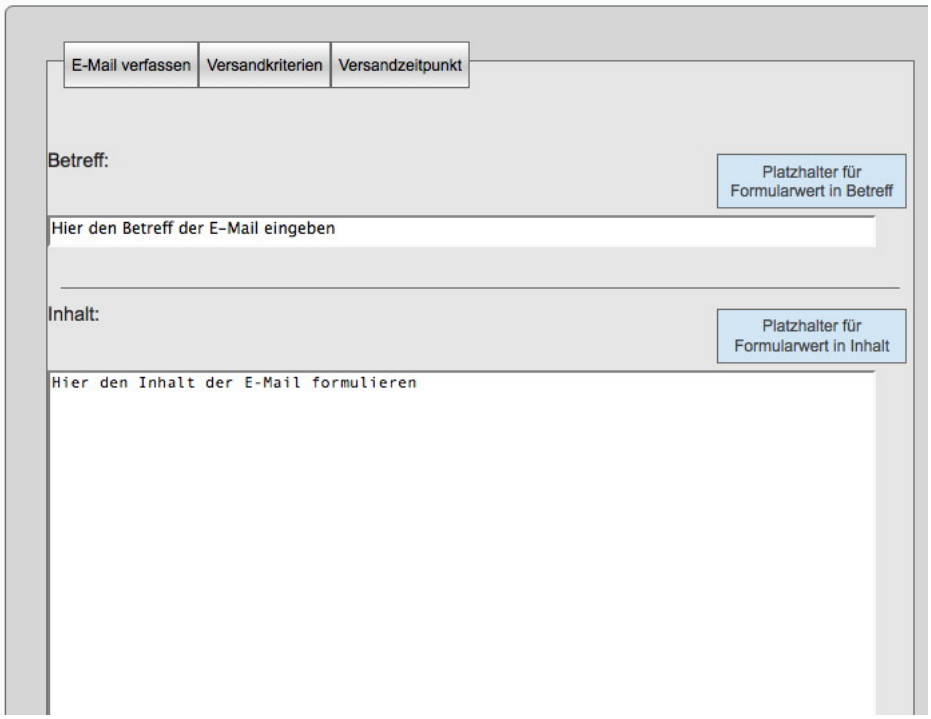


Abbildung 10.5: *perForm*: Verfassen der E-Mail-Bestätigung an den Benutzer des Formulars

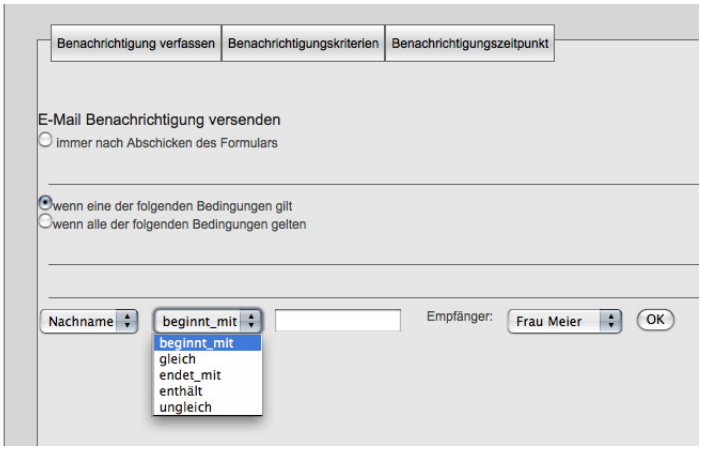


Abbildung 10.6: *perForm*: Einstellen der Benachrichtigungskriterien für die internen E-Mail-Benachrichtigungen

Für die als kritisch identifizierten Mikrotasks sollen in den folgenden Abschnitten die aufgetretenen Probleme genauer untersucht werden.

Aus Gründen der Übersichtlichkeit seien hier noch einmal die in Abschnitt 5.3 eingeführten Problemkategorien aufgeführt:

Orientierunglosigkeit im Modellierungsprozess	(OrientLosModProz)
Suche nach Bedienelement	(SuchBedElem)
Unsicherheit ob richtiges Bedienelement gewählt	(UnsicherhRichtBedElem)
Unsicherheit über Richtigkeit der durchgeführten Aktion	(UnsicherhRichtAktion)
Verunsicherung durch zahlreiche Auswahlmöglichkeiten	(VerunsZahlrAusw)
Probleme in der Bedienung des Layouts	(ProbBedLay)
(Bedien-) Konzept unklar	(KonzUnklar)

Auch die möglichen Ursachen für Usability-Probleme in diesem Kontext seien hier noch einmal genannt:

1 keine/schlechte Führung durch den Gesamtprozess

2 Konzept/Aufbau der Software unklar

3 fehlendes Feedback

4 fehlende Information

5 schlechte Strukturierung der Oberfläche

6 schlechte Bedienbarkeit

7 nicht aussagekräftige Bezeichnungen/Icons

8 Überforderung durch Komplexität

9 inkonsistentes Bedienungskonzept

In Tabelle 10.2 wird für jedes aufgetretene Usability-Probleme dessen Zuordnung zu einer Problemkategorie und Ursache vorgenommen.

Mikrotask	Problem	Häufigkeit	Problemkategorie	Ursache
2	Der Benutzer sucht, wo ein neues Formular zu erstellen ist.	3	SuchBedElem	7
	Es ist dem Benutzer nicht klar, dass das default-Layout bereits ausgewählt ist	1	OrientLosModProz	4
	Der Benutzer findet es unklar, was er im ersten Schritt ausgewählt hat.	1	OrientLosModProz	3
18	Dem Benutzer ist unklar, welches Formularelement zu wählen ist	2	UnsicherhRichtBedElem	7
22	Der Benutzer sucht, wo die Anzeigeoptionen für ein Formularelement einzustellen sind	2	SuchBedElem	5
26	Der Benutzer gibt die Vorbelegung in die Eigenschaft 'interne Beschreibung' ein	1	SuchBedElem	7
27	Der Benutzer muss sich in Modellierungsprozess orientieren um Bedienelement für die E-Mail-Bestätigung zu finden	3	OrientLosModProz	2
29	Der Benutzer sucht in den Versandkriterien und probiert Regeldefinitionen durch, erkennt aber nicht, dass er bereits an der richtigen Stelle ist	1	SuchBedElem	6
33	Der Benutzer schaut in den Verarbeitungskriterien, erkennt aber nicht, dass er bereits an der richtigen Stelle ist.	1	SuchBedElem	6

Tabelle 10.2: *perForm*: Aufgetretene Usability-Probleme

10.2 Anzahl der Usability-Probleme pro Problemkategorie

Um eine Aussage über das Ausmaß der jeweils in den kritischen Mikrotasks aufgetreten Probleme treffen zu können, wird mithilfe von Tabelle 10.3 für jeden Mikrotask analysiert, zu wie vielen Usability-Problemen es in welcher Problemkategorie kam. Dabei wird für jedes Usability-Problem die Häufigkeit des Auftretens bei allen Probanden berücksichtigt.

Eine Zuordnung der aufgetretenen Usability-Probleme zu den einzelnen Problemkategorien ergibt die in Tabelle 5.8 gezeigte Verteilung.

Mikrotask	OrientLosModProz	SuchBedElem	UnsicherhRichtBedElem	UnsicherhRichtAktion	VerunsZahlrAusw	ProbBedLay	KonzUnklar	Summe
2	2	3	0	0	0	0	0	5
18	0	0	2	0	0	0	0	2
22	0	2	0	0	0	0	0	2
26	0	1	0	0	0	0	0	1
27	3	0	0	0	0	0	0	3
29	0	1	0	0	0	0	0	1
33	0	1	0	0	0	0	0	1
Summe		5	8	2	0	0	0	15

Tabelle 10.3: *perForm*: Aufgetretene Usability-Probleme pro Mikrotask, gruppiert nach Problemkategorie

In der folgenden Auflistung sollen nun die kritischen Mikrotasks näher betrachtet und analysiert werden.

I Die meisten Usability-Probleme traten bei der Durchführung von Mikrotask 2 auf, bei dem es darum ging, ein neues Formular mit dem default-Layout zu erstellen. Die Schwierigkeit für die Probanden war an dieser Stelle, dass sie nicht erkennen konnten auf welche Weise sie ein vorhandenes Layout auswählen können, da sie das entsprechende Bedienelement nicht als solches wahrnahmen. Dies ist darauf zurückzuführen, dass das entsprechende Bedienelement nicht deutlich und aussagekräftig genug bezeichnet und schlecht wahrnehmbar war. Auch war den Probanden nach Auswahl des Layouts nicht ersichtlich, dass sie nun ein Layout ausgewählt haben und mit der Bearbeitung der Formularelemente beginnen können. Diese Information wurde den Probanden von der Anwendung nicht ausdrücklich gegeben, ebenso fehlte den Probanden diesbezüglich das visuelle Feedback.

II Auch verhältnismäßig häufig kam es zu Problemen in Mikrotask 27. Dort war von den Probanden gefordert, die E-Mail-Bestätigung an den Benutzer des Formulars zu aktivieren. Hierbei brauchten die Probanden eine gewisse Zeit, um sich im Modellierungsprozess zu orientieren und zu erkennen, dass das Bearbeiten der E-Mail-Bestätigungen ein separater Schritt des Modellierungsprozesses ist.

III Relativ wenige Probleme traten bei Mikrotask 18 auf. Hier bestand das Problem darin, dass die Bezeichnung des Bedienelements für die Probanden nicht eindeutig war. Dadurch waren die Probanden unsicher, ob sie das richtige Bedienelement ausgewählt haben.

IV Gleichermaßen wenige Probleme gab es bei der Durchführung von Mikrotask 22, bei dem die Probanden nicht sofort das Bedienelement zum Einstellen der Anzeigeoptionen eines Formularelements fanden. Die Anzeigeoptionen für ein Formularelement sind unterhalb des Eigenschaften-Inspektors angeordnet und fielen nicht sofort in das Blickfeld der Probanden.

V In den Mikrotasks 26, 29 und 33 bestand vereinzelt das Problem, dass die Probanden das Bedienelement zur Durchführung ihrer Aktion nicht sofort fanden.

10.3 Gruppierung der Usability-Probleme nach Problemkategorie

Nachdem die einzelnen Usability-Probleme bezüglich ihrer Problemkategorien und Ursachen betrachtet wurden, ist es nun interessant, zu betrachten in welchen Problemkategorien die Usability-Probleme hauptsächlich auftauchten (vgl. Abschnitt 5.3.1.2 und 5.3.2.2). Dieses entspricht der Auswertung der letzten Zeile in Tabelle 10.3 und ermöglicht Aussagen über die noch problematischen Bereiche der entwickelten Methode.

Fasst man also die aufgetretenen Usability-Probleme nach Problemkategorien zusammen, so ergibt sich die in Abbildung 10.7 gezeigte Verteilung.

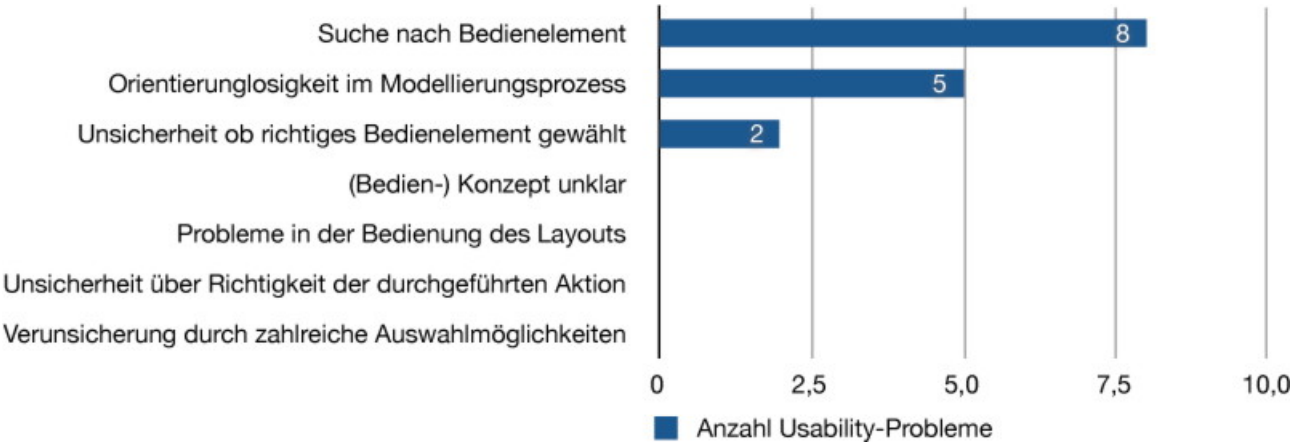


Abbildung 10.7: *perForm*: Summe der aufgetretenen Usability-Probleme pro Problemkategorie

In Abbildung 10.7 zeigt sich deutlich, dass die meisten Probleme in der Problemkategorie 'Suche nach Bedienelement' auftraten. Die Anwendung der entwickelten Methode hat somit noch relativ große Defizite bzgl. der aussagekräftigen Bezeichnung von Bedienelementen und teilweise der Bedienbarkeit. Diese Probleme in der Bedienbarkeit liegen hauptsächlich darin, dass der für die Usability-Tests verwendet Prototyp noch keine visuell ausgereiftes Produkt repräsentiert. Es gibt noch keine vollkommen konsistente Darstellung und Positionierung der Bedienelemente, was jedoch entscheidende Usability-Faktoren sind.

Auch wenn den Probanden anhand der oberen Navigation insgesamt klar war, dass sie einen schrittweisen Modellierungsprozess durchlaufen, gab es doch zum Teil Probleme bzw. Verzögerungen in der Orientierung innerhalb dieses Prozesses. Dies war insbesondere an Stellen der Fall, wo dem Probanden seitens der Anwendung keine Information oder kein Feedback über seine letzte Aktion gegeben wurde, wie beispielsweise der Auswahl eines Formular-Layouts. Dadurch war der Proband über den Erfolg seiner Aktion und seinen Status im Modellierungsprozess verunsichert. Zu Verzögerungen bei der Orientierung kam es bei dem

Aktivieren der E-Mail-Bestätigung, da dieses durch einen dedizierten Schritt in der Anwendung repräsentiert ist und dieses Konzept erst vom Probanden erfasst werden musste.

Vereinzelt waren die Probanden verunsichert, ob sie das für ihre Aktion richtige Bedienelement gewählt haben, da sie dieses der Darstellung bzw. Bezeichnung des entsprechenden Bedienelements nicht entnehmen konnten.

10.4 Auswertung der Interviews

Neben der Auswertung der während der Testdurchführung erhobenen Daten, welche in den vorangegangenen Abschnitten erfolgte, liefern auch die nach den Tests durchgeführten Interviews mit dem Probanden relevante Informationen zur Gebrauchstauglichkeit der entwickelten Anwendung. In den Interviews schildern die Probanden ihre Impressionen, die sie während der Arbeit mit der entwickelten Methode hatten. Dadurch erhält man Informationen über die User-Experience der Probanden mit der Anwendung, welche die verschiedensten Aspekte der Eindrücke der Probanden bei der Interaktion mit der Anwendung umfasst. Darüber hinaus erhält man ein sehr gutes Bild der positiven und negativen Aspekte der Anwendung sowie Informationen über Verbesserungspotentiale.

Die Probanden reflektierten allgemein, dass sie die obere Navigationsleiste, welche die einzelnen Schritte repräsentiert, sehr gut als Navigationsmöglichkeit innerhalb des Modellierungsprozesses wahrnahmen und ihnen dadurch auch klar war, dass sie sich in einem schrittweisen Prozess befinden. Wie auch in Abschnitt 5.3.1.3 war das Drag-and-Drop Bedienungskonzept für die Probanden einfach und intuitiv und es gab im Gegensatz zu der Testdurchführung mit dem Produkt *jotform* keine Probleme in dessen Benutzung. Das Bedienungskonzept zum Hinzufügen und Bearbeiten von Formularelementen wurde durchweg als sehr einfach und intuitiv angesehen.

Als **negative Aspekte** nannten die Probanden die folgenden Punkte:

- Die Bezeichnungen der Bedienelemente, das so genannte Wording, war zum Teil nicht intuitiv.
- Beim Drag-and-Drop der Formularelemente wurde es als verwirrend empfunden, dass beim Drop, also dem ‘Fallenlassen’ des Elements über dem Formular, der Revert-Effekt eingesetzt wird, obwohl das Element dem Formular hinzu gefügt wird. Beim Revert-Effect ‘fliegt’ das Icon nach dem Drop zurück zu seiner Drag-Position. Hier kam es zu Irritationen, da der Revert-Effekt den Probanden suggerierte, dass die Drop-Aktion gescheitert ist, obwohl dieses nicht der Fall war.
- Die Bedeutung und somit auch der Unterschied zwischen denen mit ‘Bestätigung’ und ‘Verarbeitung’ betitelten Schritten war den Probanden nicht intuitiv klar.

- Der Sinn und Zweck der Eigenschaft ‘interne Beschreibung’ hat sich den Probanden häufig nicht sofort erschlossen.
- Die Bezeichnung ‘Validierung’ wurde teilweise als zu technischer Begriff angesehen.

Als **positive Aspekte** stellten sich die folgenden Punkte in den Interviews heraus:

- Die Oberfläche wurde als gut strukturiert und sehr übersichtlich wahrgenommen und verwirrte nicht durch unangemessene Komplexität.
- Die meisten Bedienelemente waren leicht zu finden.
- Das Produkt ist in sich schlüssig und beinhaltet die für die Arbeitsdomäne der Probanden notwendigen Funktionen.
- Die Probanden hatten weitgehend einen Überblick über den Gesamtprozess und sahen sich in ihrer Bearbeitungsreihenfolge der einzelnen Schritte zwar geführt, jedoch empfanden sie diese nicht als aufgezwungen.
- Das sofortige visuelle Feedback im WYSIWYG-Formular während der Bearbeitung der Eigenschaften der Formularelemente gab den Probanden ein Gefühl der Sicherheit über die Richtigkeit ihrer durchgeführten Aktionen.

Verbesserungspotential sahen die Probanden in den folgenden Punkten:

- Auf der Palette mit den einzufügenden Formularelementen sollten häufig verwendete Elemente wie Texteingabefelder auch entsprechend weit oben angeordnet sein, und nicht mittig oder im unteren Bereich. Abbildung 10.8 illustriert diesen Aspekt.
- Im Schritt der Layoutauswahl sollten dem Benutzer Formularvorlagen zur Verfügung gestellt werden, die er auswählen und anschließend nach Belieben bearbeiten oder erweitern kann.
- Jeder Schritt des Modellierungsprozesses sollte eine Art ‘weiter’ Button enthalten, über den der Benutzer die Bearbeitung des aktuellen Schrittes abschliessen und zum nächsten Schritt übergehen kann. Dabei sollen dem Benutzer entsprechende Hinweise über ungültige oder fehlende Eingaben gegeben werden, welche zum Abschluss des aktuellen Schrittes noch notwendig sind.
- Es sollte mehr und ausführlicher mit Hinweistexten gearbeitet werden, die den Benutzern bestimmte Konzepte oder Vorgehensweisen erläutern.
- Innerhalb des Eigenschaften-Inspektors sollte eine Tab-Navigation zwischen den Eigenschaften möglich sein.

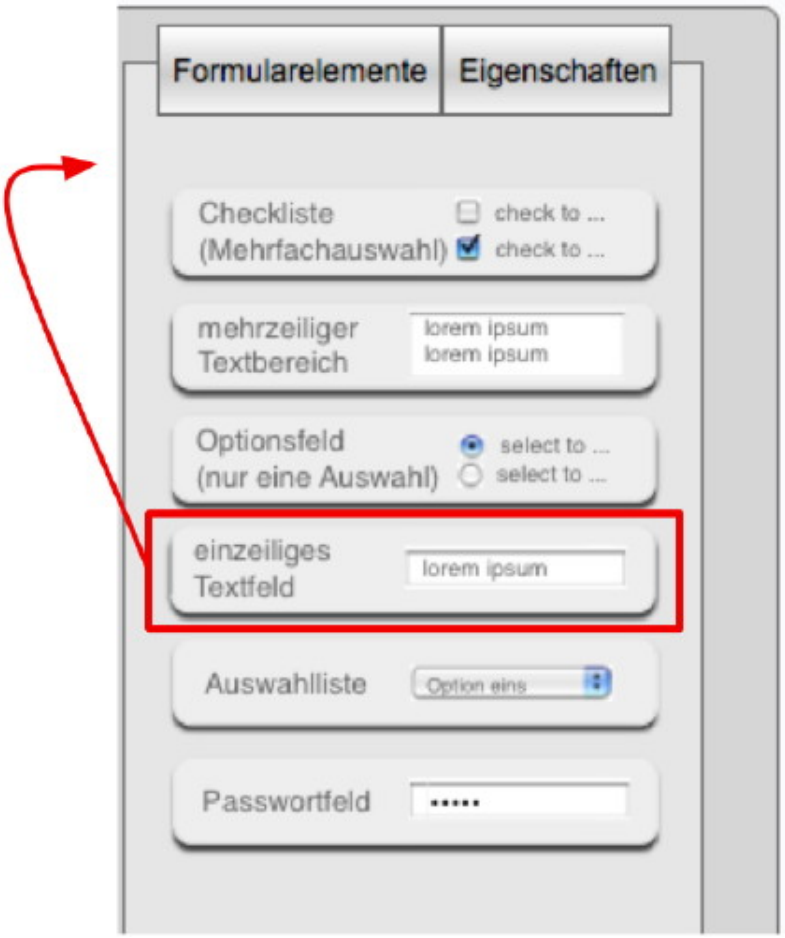


Abbildung 10.8: *perForm*: Ansicht des Bedienelements mit den einzufügenden Formularelementen

11 Fazit

Die im Dialog mit den Domänenexperten durchgeführte Evaluierung und anschließende Analyse bestehender Systeme für die Domäne der Modellierung von Prozessen in Webportalen hat Aufschluss über die Stärken und Schwächen der verschiedenen Produktansätze gegeben. Auch zeichneten sich bestimmte Präferenzen der Domänenbenutzer in diesem Bereich ab.

Basierend auf den Ergebnissen der Analyse und unter Betrachtung der Benutzerpräferenzen wurde eine Methode zur Verbesserung der mit den vorhandenen Systemen identifizierten Usability-Problemen entwickelt. Dabei bestand der wesentliche Punkt darin, die Ursachen der evaluierten negativen Aspekte zu erkennen und anhand dessen Überlegungen und Abwägungen zu alternativen Gestaltungs- und Konzeptmöglichkeiten vorzunehmen. Dazu waren zum einen theoretische Betrachtungen auf Grundlage der verschiedenen in Abschnitt 2.2 erläuterten mentalen Aspekte der Programmierung und allgemeiner Usability-Richtlinien notwendig. Zum anderen gab es auch während dieser Phase der theoretischen Betrachtungen immer wieder den Dialog mit den Domänenexperten, um Feedback bzgl. einzelner Konzepte und Gedankengänge zu erhalten.

Aus diesen Betrachtungen entstand unter Einbeziehung der als positiv evaluierten Usability-Aspekte und der erkannten Benutzerpräferenzen eine Methode, welche die negativen Usability-Aspekte der betrachtete Produkte beheben sollte. Um die theoretischen Überlegungen verifizieren zu können und feststellen zu können, ob mit dieser entwickelten Methode Verbesserungen bestimmter Usability-Aspekte gegenüber den bestehenden Softwareprodukten erreicht werden konnten, musste diese Methode zunächst als Prototyp umgesetzt werden. Dies erfolgte in Form einer mit dem Namen *perForm* bezeichneten Webanwendung.

Anhand des implementierten Prototypen konnten dann, analog zur Evaluierung der bestehenden Softwareprodukte, Usability-Tests mit der Anwendung *perForm* und den Domänenexperten erfolgen. Die Auswertung und Analyse erfolgte ebenfalls analog zu dem bei der Evaluierung der bestehenden Softwareprodukte angewendeten Vorgehen.

Bei einem Vergleich der Ergebnisse der Tests mit den bestehenden Softwareprodukten und der Anwendung *perForm* ist zu berücksichtigen, dass es sich bei der Anwendung *perForm* um einen Prototypen handelt. Da die funktionale Unvollständigkeit des Prototypen nicht zu einer Verfälschung der Testergebnisse führen sollte, wurde eine bzgl. der Bedienelemente möglichst vollständige Benutzeroberfläche geschaffen. Dies ist jedoch nur in einem begrenzten Rahmen möglich. Eine Weiterentwicklung des Prototypen und das Hinzufügen sich

ändernder oder neuer Anforderungen wird ein Anwachsen der Komplexität in der Benutzeroberfläche nach sich ziehen. Entsprechend muss eine Weiterentwicklung der Anwendung von weiteren Usability-Tests begleitet werden, um eine gute Usability der Anwendung aufrecht zu erhalten bzw. zu gewährleisten.

Der Prototyp stellt jedoch alle für die Domänenexperten relevanten Funktionen dar und setzt diese insofern um, dass in den Usability-Tests funktional gleichwertige Aufgaben vom gleichen Umfang wie mit den bestehenden Produkten bearbeitet werden konnten. Abbildung 11.1 ist zu entnehmen, dass es dabei zu einer deutlich geringeren Anzahl von Usability-Problemen kam.

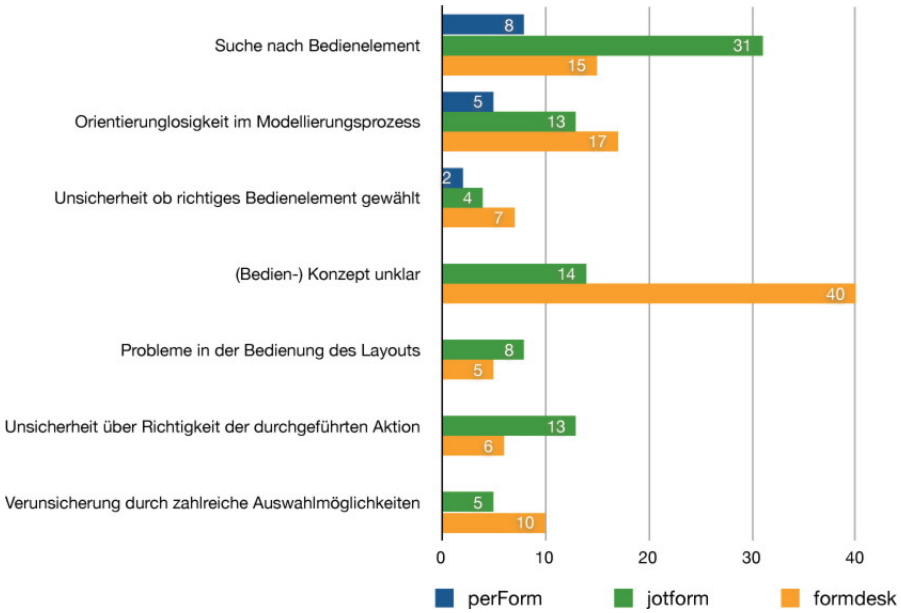


Abbildung 11.1: Summe aller in Usability-Tests aufgetretener Probleme pro Problemkategorie

Mit der Anwendung *perForm* kam es zu keinen Problemen bei denen die Probanden durch Komplexität in der Oberfläche oder durch fehlendes Feedback zu ihren Aktionen verunsichert waren. An dieser Stelle hat das beim Entwurf der Benutzeroberfläche bewusst und konsequent eingesetzte Vorgehen der klaren Strukturierung und der Komplexitätsreduktion der Oberfläche zu einer deutlich besseren Usability als bei den betrachteten Produkten geführt.

Auch bestand bei den Probanden an keiner Stelle das Problem, dass ihnen ein Konzept in der Bedienung nicht klar war oder sie mit der Bedienung des Layouts nicht zurecht kamen. Hier hat sich der Einsatz eines funktionierenden und vom Benutzer präferierten Drag-and-Drop Bedienungskonzepts positiv ausgewirkt. Ebenso zeigte sich, dass das Zurückgreifen auf die dem Benutzer geläufigen Konzepte, wie es beispielsweise bei der Definition von Bedingungen umgesetzt wurde, und die Vorgabe möglicher Eingabewerte durch das System,

um den Benutzer vor Falscheingaben zu bewahren, zu einer sehr positiven User-Experience geführt haben.

Die Probanden erkannten, dass sie sich in einem schrittweisen Prozess befinden und fanden diesen Ansatz sehr hilfreich und hatten eine gute Orientierung, an welcher Stelle des Modellierungsprozesses sie sich befinden. Probleme diesbezüglich gab es jedoch insofern, dass den Probanden der Unterschied zwischen den E-Mail-Bestätigungen, die sie an die Benutzer des Formulars verfassen können und den E-Mail-Berichten, die der internen Verarbeitung der Formulardaten dienen sollen, nicht klar war. An dieser Stelle muss die Unterscheidung der beiden Konzepte dem Benutzer klarer vermittelt werden. Hier sollte zur Verbesserung der Usability die Bezeichnung der Schritte deren Zweck klarer bezeichnen und es könnte auch an dieser Stelle mit Hinweisen gearbeitet werden. Wie in Abschnitt 2.3 für die textuelle Notation erwähnt, kann durch Hinweise sehr einfach die Zugänglichkeit zu bestimmten Informationen und Konzepten verbessert werden, was die Probanden auch im Zusammenhang mit dem in Schritt 2 erläuterten Bedienungskonzept als sehr hilfreich empfanden.

Auch sollte der Prototyp bezüglich der konsistenten Verwendung von Farben und Bedienelementen vervollständigt werden, um so die in den Bereichen ‘Suche nach Bedienelement’ und ‘Unsicherheit ob richtige Bedienelement gewählt’ aufgetretenen Usability-Probleme weiter zu reduzieren.

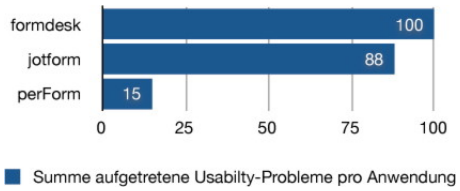


Abbildung 11.2: Summe aufgetretener Usability-Probleme pro Anwendung

Abbildung 11.2 zeigt, dass bei gleichwertigen Aufgabenstellungen für alle getesteten Produkte, die Anzahl der aufgetretenen Probleme in den Usability-Tests für die bestehenden Produkte ungefähr im gleichen Bereich lagen. In den Testdurchführungen mit der Anwendung *perForm* reduzierte sich die Anzahl der aufgetretenen Probleme um ca. 85% gegenüber den bestehenden Softwareprodukten. Dieses Ergebnis konvergiert mit den Aussagen der Probanden, dass sie zwar an einigen und im letzten Abschnitt bereits diskutierten Punkten noch Verbesserungspotential der entwickelten Methode sehen, sie mit der Anwendung *perForm* jedoch eine positive User-Experience hatten und diese insgesamt als sehr intuitiv und leicht bedienbar empfanden.

Literaturverzeichnis

[ade] *Adeptia BPM Server*. http://www.adeptia.com/products/products.html.

[ari] *Aris Business Designer*. http://www.ids-scheer.de/de/ARIS/ARIS_Software/ARIS_Business_Designer/7773.html.

[Ast03] ASTELS, DAVE: *Test-Driven Development: A Practical Guide*. Prentice Hall International, 2003.

[biz] *BizFlow*. http://www.adeon.ch/produkte/bizflow.php.

[Ble] BLEIKER, ANDREAS: *Mit wie vielen Anwendern testet man bei einem Usability-Test?* http://www.usability.ch/News_D_Textonly/T_hm_user.htm.

[bre] *Breadcrumb Navigation Increasingly Useful*. http://www.useit.com/alertbox/breadcrumbs.html.

[Coh04] COHN, MIKE: *User Stories Applied For Agile Software Development*. Addison Wesley, 2004.

[Dav] DAVIES, S.P.: *Externalizing Information During Coding Activities: Effects of Expertise, Environment and Task*. .

[Eica] EICHINGER, ARMIN: *Cognitive Walkthrough*. http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Doktoranden/absolventen/eichinger_armin/u-walkthrough.html.

[Eicb] EICHINGER, ARMIN: *Heuristische Evaluation*. http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Doktoranden/absolventen/eichinger_armin/u-evaluation.html.

[Eicc] EICHINGER, ARMIN: *Usability Inspection*. http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Doktoranden/absolventen/eichinger_armin/u-inspection.html.

[Eicd] EICHINGER, ARMIN: *Usability Tests*. http://www.uni-regensburg.de/Fakultaeten/phil_Fak_II/Psychologie/Doktoranden/absolventen/eichinger_armin/u-tests.html.

[fora] *FormLogix*. http://www.formlogix.com/CreateWebForms.aspx.

[forb] *Formular Generator*. http://www.formular-generator.de/.

[Fro] FROMMANN, UWE: *Die Methode 'Lautes Denken'*. http://www.e-teaching.org/didaktik/qualitaet/usability/Lautes%20Denken_e-teaching_org.pdf.

[geb] *Gebrauchstauglichkeit (Produkt)*. http://de.wikipedia.org/wiki/Gebrauchstauglichkeit_(Produkt).

[Gre96] GREEN, T. R. G., PETRE M.: *Usability Analysis of Visual Programming Environments: a 'cognitive dimensions' framework*. , MRC Applied Psychology Unit, / Dept. of Mathematics and Computer Science, Open University, 1996. http://homepage.ntlworld.com/greenery/workStuff/Papers/UsabilityVPs.PDF.

[ibm] *WebSphere Process Server*. http://www-142.ibm.com/software/dre/ecatalog/detail.wss?locale=de_DE&synkey=L106045O85235S77.

[ipr] *TIBCO iProcess Suite*. http://www.tibco.com/solutions/bpm/iprocess_suite.jsp.

[Joh00a] JOHNSON, JEFF: *GUI Bloopers*, 24–25. [Joh00b], first , 2000.

[Joh00b] JOHNSON, JEFF: *GUI Bloopers*. Morgan Kaufmann, first , 2000.

[J.W] J.WEBSTER: *Visual Programming*. ftp://ftp.cs.orst.edu/pub/burnett/whatIsVP.pdf.

[kog] *Cognitive Walkthrough*. http://www.bui.haw-hamburg.de/pers/ursula.schulz/webusability/methcw.html.

[Koh] KOHL, LUTZ: *Konzepte der visuellen Programmierung und ihrer Einsatzmöglichkeiten an Schulen*. www.uni-jena.de/data/unijena_/faculties/minet/casio/LutzKohl/StudienarbeitLutzKohl.pdf.

[mac] *Webformulare und Ergebnisübersicht*. http://www.appnitro.com/.

[met] *Methoden zur Usability-Testung*. http://www.e-teaching.org/didaktik/qualitaet/usability/.

[Nie94] NIELSEN, JAKOB: *Heuristic evaluation*. John Wiley and Sons, New York, 1994.

[NL93] NIELSEN, JAKOB THOMAS K. LANDAUER: *A mathematical model of the finding of usability problems*. *CHI '93: Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, 206–213, New York, NY, USA, 1993. ACM.

[pag] *PageBreeze Free HTML Editor*. http://www.pagebreeze.com/.

[pfo] *Online-Formular Generator pForm*. http://www.phpform.org/.

[Ram] RAMPL, DIPL. INF (FH) HANSJÖRG: *Definition Usability*. http://www.handbuch-usability.de/begriffsdefinition.html.

[sav] *Savvion Process Modeler*. http://www.savvion.com/products/process-modeler.php.

[Scha] SCHIFFER, STEFAN: *Visuelle Programmierung - Potential und Grenzen*. http://www.schiffer.at/publications/se-96-19/se-96-19.pdf.

[Schb] SCHMIDT, CARSTEN: *Visuelle Sprachen und Ihre Ausdrucksmittel*. ag-kastens.upb.de/lehre/paderwave/material/cschmidt-praesentation.pdf.

[Sim] SIMON, ANDREAS: *Usability Testing - Theorien, Modelle und Methoden der Softwareevaluation*. www.wi.uni-muenster.de/pi/lehre/ws0809/SeminarSE2/ausarbeitungen/UsabilityTesting.pdf.

[SS86] SPOHRER, JAMES C. ELLIOT SOLOWAY: *Novice mistakes: are the folk wisdoms correct?* Commun. ACM, 29(7):624–632, 1986.

[Str] STROHMAIER, ROBERT: *Dokumentation / Beschreibung – Usability Tests Thinking Aloud Testing*. http://dmt.fh-joanneum.at/projects/ebus2/objects/application_pdf/Test_Doku_ThinkingAloud_v1.pdf.

[swe] *Software-Ergonomie*. http://de.wikipedia.org/wiki/Software-Ergonomie.

[swq] *ISO/IEC 9126*. http://de.wikipedia.org/wiki/ISO_9126.

[T.R] T.R.G.GREEN, M. PETRE: *Usability Analysis of Visual Programming Environments: a 'cognitive dimensions' framework*. homepage.ntlworld.com/greenery/workStuff/Papers/UsabilityVPs.PDF.

[wor] *WorkflowXPert.NET Dokumentation*. http://www.workflowxpert.net/Default.aspx?tabid=158#FL.

[Wul] WULF, PROF. DR. VOLKER: *End User Development Definition*. https://wiki.wineme.fb5.uni-siegen.de/bin/view/DeutschesEUDWiki/EndUserDevelopment?TWIKISID=72ef36050c766bfb0c09c94361e814d1.

Anhang A

Kategorisierte Übersicht der betrachteten Produkte

Produkte der Kategorie Formulargeneratoren:

- **PageBreeze**: HTML Editor, unterstützt zur Darstellung die beiden Modi WYSIWYG und HTML-Quellcode. Features: Formularerstellung per Drag and Drop, Vorschaufunktion, integrierte Schnittstelle zum angebotenen Formular-Verarbeitungsservice [pag].
- **FormLogix**: online WYSIWYG Formulargenerator. Features: Formularerstellung per Drag and Drop, zur Einbindung des Formulars bekommt der Benutzer ein Skript zugeschickt, dass er dann Kopieren und in seine Website einbinden kann, Formular-Hosting: Benachrichtigung bei eingegangenen Daten, Übersicht und Analyse der eingegangenen Formulardaten [fora].
- **jotform**: WYSIWYG Formulargenerator. Features: Formularerstellung per Drag and Drop, Formulareinbindung per Quelltext oder Verlinkung zu gehostetem Formular, Formular-Hosting: E-Mail-Benachrichtigung bei eingegangenen Formulardaten, Übersicht der eingegangenen Formulardaten
- **FormularGenerator**: online Formular-Generator. Zur Formulareinbindung wird nach dem Erstellen des Formulars der fertige Quellcode per E-Mail zugeschickt [forb]
- **4Webmaster.net**: online Formulargenerator mit Assistent, der am Ende das Formular als PHP-Datei zum Download bereit stellt, Felder müssen nacheinander definiert werden, bietet keine Vorschau des Formulars und keine Designmöglichkeiten. [forb] Screenshot siehe Anhang C B.5.
- **pForm**: Online-Formular Generator der in drei Schritten durch den Erstellungsprozess führt, im ersten Schritt kann der ABenutzer ein vorhandenes Layout aussuchen, im zweiten Schritt die Formularelemente hinzufügen und im dritten Schritt kann der Quelltext herunter geladen werden. Das Versenden von E-Mails ist nicht möglich. [pfo]

Produkte der Kategorie Geschäftsprozessmodellierung:

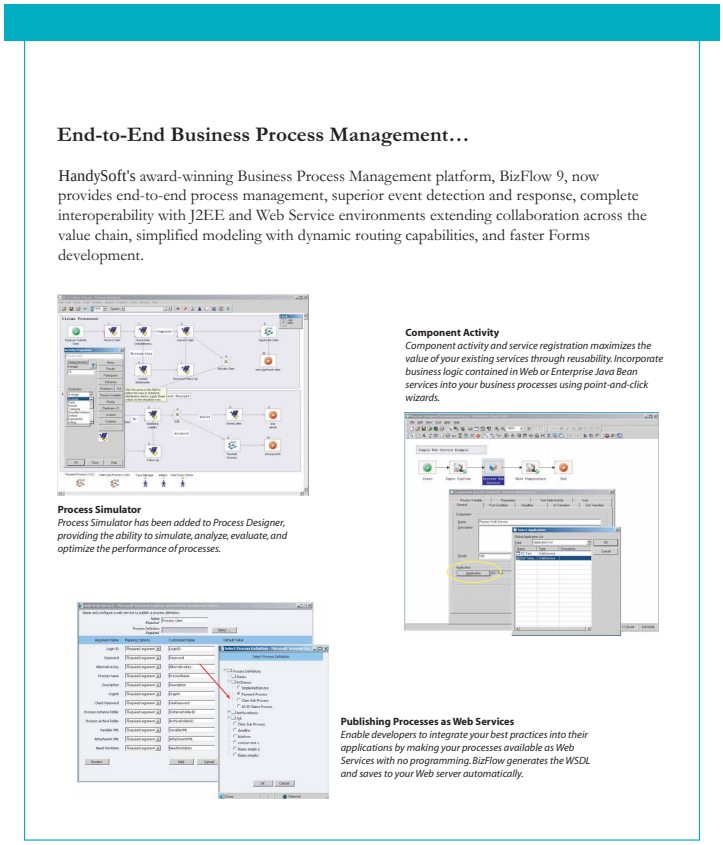
- **WorkflowXPert.NET**: Workflow Management System (WFMS) für die automatisierte Verarbeitung von individuellen Geschäftsprozessen jeglicher Art [wor]. Für die Abbildung der Prozesse wird eine visuelle, datenflussorientierte Modellierung verwendet (siehe Anhang C B.2).
- **ARIS Business Architect**: Web-basierte Software für ein professionelles, unternehmensweites Business Process Management (BPM). Modellierung, Analyse und Optimierung von Geschäftsprozessen [ari].
- **BizFlow**: Vereinfachung und Automatisierung von Geschäftsprozessen [biz]. Auch hier erfolgt eine grafische Modellierung der Prozesse, Screenshots finden sich im Anhang C.
- **TIBCO iProcess Suite**: Handhabung jeglicher Art von Prozessen über den gesamten Prozess-Lebenszyklus [ipr].
- **Savvion Business Manager**: BPM Suite [sav]. Ein Screenshot der im Process Modeler verwendete grafischen Modellierung der Prozessabläufe findet sich im Anhang C (B.3)
- **IBM WebSphere Business Modeler**: Geschäftsprozesse vollständig visualisieren, nachvollziehen und dokumentieren [ibm].
- **Adeptia BPM Server**: Managen, Automatisieren und Optimieren von Geschäftsprozessen [ade].

Betrachtete Produkte der Kategorie Formular-Management Systeme:

- **MachForm**: Browser-basierter, selbst zu hostender HTML-Formulargenerator. Features: Formularerstellung per Drag and Drop, Formularmanager zur Verwaltung aller Formulare, E-Mail-Benachrichtigungen, auch an mehrere E-Mail-Adressen, Formulardaten werden in einer Datenbank gespeichert, Integration in bestehendes Design möglich [mac]
- **formdesk**: Formular-Management-System. Features: personalisierte E-Mail-Benachrichtigungen, individuelle Rechte für Personen und Abteilungen, Formdesk - Workflow: Digitale Verarbeitung der online ausgefüllten Formulare, Hosting der Formulare: Formulare und empfangene Daten können in verschiedenen Datenformaten heruntergeladen werden.

Anhang B

Screenshots betrachteter Produkte



End-to-End Business Process Management…

HandySoft's award-winning Business Process Management platform, BizFlow 9, now provides end-to-end process management, superior event detection and response, complete interoperability with J2EE and Web Service environments extending collaboration across the value chain, simplified modeling with dynamic routing capabilities, and faster Forms development.

Process Simulator
Process Simulator has been added to Process Designer, providing the ability to simulate, analyze, evaluate, and optimize the performance of processes.

Component Activity
Component activity and service registration maximizes the value of your existing services through reusability. Incorporate business logic contained in Web or Enterprise Java Bean services into your business processes using point-and-click wizards.

Publishing Processes as Web Services
Enable developers to integrate your best practices into their applications by making your processes available as Web Services with no programming. BizFlow generates the WSDL and saves to your Web server automatically.

Abbildung B.1: BizFlow Formulargenerator, Quelle: http://www.bizflow/formulargenerator.htm

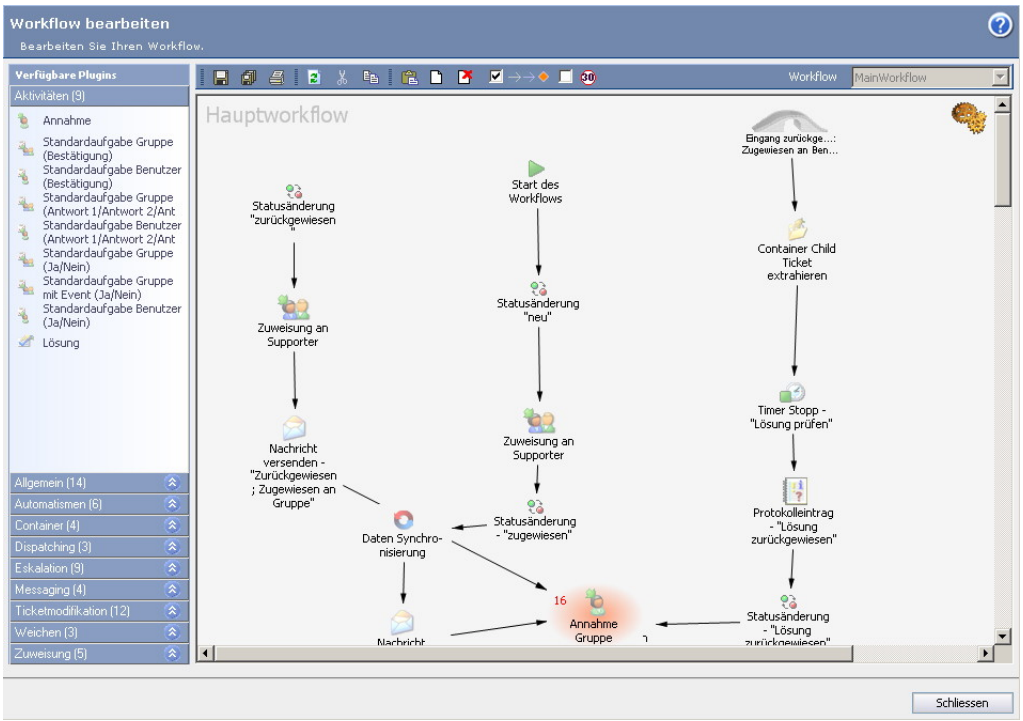


Abbildung B.2: WorkflowXPert, Quelle: http://www.workflowxpert.net/Default.aspx?tabid=158

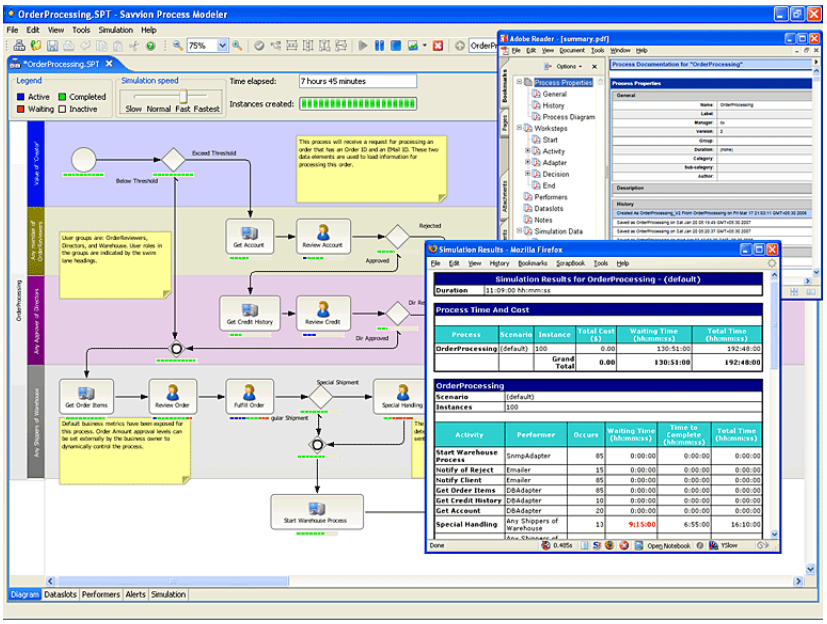


Abbildung B.3: Savvion Process Modeler, Quelle: http://www.savvion.com/products/process-modeler.php

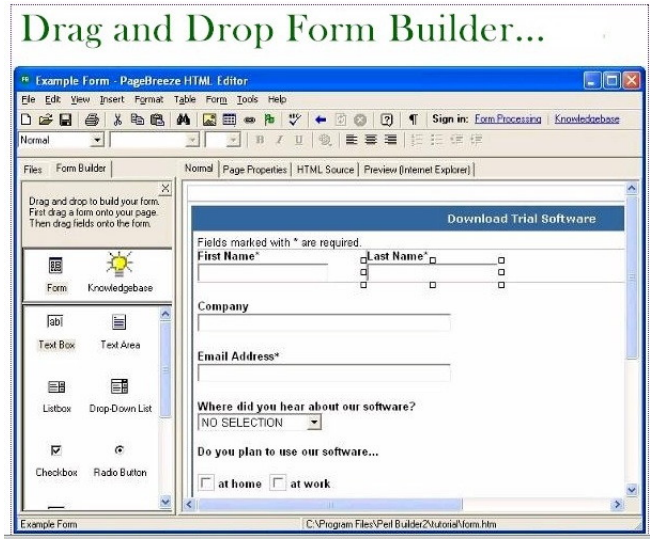


Abbildung B.4: PageBreeze HTML Editor, Quelle: http://www.pagebreeze.com/

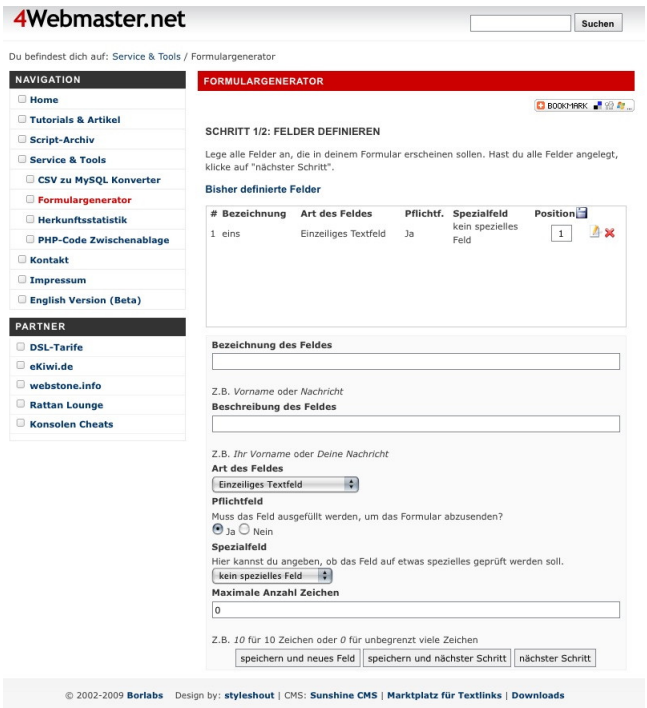


Abbildung B.5: 4Webmaster Formulargenerator, Quelle: http://www.4webmaster.net/service_tools/formulargenerator.htm

Anhang C

Der Testplan

Freie Universität Berlin
Fachbereich Informatik
AG Software Engineering

infopark
Research and
Development

„Thinking Aloud“ Testplan

Inhalt

Ziel des Thinking Aloud Usability-Tests 2
Userprofil des Probanden 2
Einverständniserklärung 3
Testumgebung 4
Testprotokoll 5
Testdurchführung jotform Formulargenerator 6
Testdurchführung: formdesk 14
Testdurchführung: WebDSL 21

Ziel des Thinking Aloud Usability-Tests

Bei Usability Tests soll die ergonomische Qualität von Software getestet bzw. verbessert werden, die Methode Thinking Aloud (lautes Denken) ist die am meisten verbreitete Usability Testmethode. Es handelt sich hierbei um ein psychologisches Experiment, das es ermöglicht die Benutzerfreundlichkeit einer Benutzeroberfläche festzustellen indem die Testpersonen ihre Eindrücke während der Benutzung der Software lautsprachlich wiedergeben. Aus den spontanen Äußerungen kann nachvollzogen werden, wie die Nutzer mit der Software interagieren

In diesem Fall soll die Benutzbarkeit verschiedener Softwareprodukte für die Modellierung von Prozessen im Web, insbesondere dem Formularwesen untersucht werden.
Dabei steht im Vordergrund herauszubekommen, wie leicht verständlich und intuitiv bedienbar diese Produkte für Domänenexperten sind.

Userprofil des Probanden

Kategorie	
Name	
Alter	
Geschlecht	
Ausbildung	
beruflicher Tätigkeitsbereich	
hauptsächliche Nutzung des Computers	
Erfahrung im Bereich Web-Programmierung	
Erfahrung mit Produkten zur Prozessmodellierung	

Einverständniserklärung

Ich stehe aus freien Stücken als Testperson in einem Usability-Test zur Verfügung, der von Julia Schenk im Rahmen ihrer Diplomarbeit im Studiengang Informatik an der Freien Universität in Zusammenarbeit mit der Infopark AG durchgeführt wird.

Ich erkläre mich einverstanden, dass die dabei erhobenen Daten in Form von Bildschirm- und Audio-Aufnahmen und schriftlich erhobenen Befragungsangaben im Rahmen von Forschung, Lehre und in Praxisfeld- bezogenen Zusammenhängen ausgewertet und verwendet werden dürfen.

_____

Datum Unterschrift

Testumgebung

Der Usability-Test wird in einem Meeting-Raum der Infopark AG durchgeführt.

Die Testdurchführung erfolgt in einer ruhigen Umgebung, damit es zu keinerlei externer Beeinflussung der Testperson kommt. Im Raum befinden sich lediglich die Testperson und ein Testadminstrator.

Equipment

Computer	Apple MacBook
Computer CPU	2.1 GHz Intel Core 2 Duo
Computer Arbeitsspeicher	2 GB
Betriebssystem	Mac OS X 10.5.6
Größe des externen Monitors	17“ TFT
verwendete Auflösung des externen Monitors	1280 x 1024
Internetanbindung	WLAN

Die Testdurchführung wird durch ein Screen-Capture mit Audio-Aufnahme aufgezeichnet.

Testprotokoll

zu testende Softwareprodukte

Bereich Softwaresuiten zur Prozess- und Workflowmodellierung
- formdesk

Bereich DSLs für Webanwendungen
- WebDSL(websdl.org)
  - WebDSL is a domain-specific language for developing dynamic web applications with a rich data model.

Bereich Formulargeneratoren
- jotform Formulargenerator
  - **Webbasierter** WYSIWYG Formulargenerator
  - Drag-and-drop Interface + Wizards
  - Features: Kontaktformulare, E-Mail Benachrichtigungen, Payment-Tools, Captcha, Bewertung, ...

Einführung

- Nicht Sie als Proband sollen getestet werden, sondern das System.
- Es geht nicht darum, wie gut oder schnell Sie eine Aufgabe lösen, sondern dass Sie bei der Bearbeitung der Arbeitsschritte ihre Eindrücke, Gefühle und Probleme wiedergeben.
- Insgesamt werden sechs auf veschiedenen Ansätzen basierende Systeme getestet.
- Das Ziel ist es die jeweiligen Stärken und Schwächen der einzelnen Systeme sowie deren Eignung für den Bereich der Modellierung von Prozessen in Web-Portelen heraus zu arbeiten.
- Der Proband bekommt die Aufgabenstellung gezeigt und kann Rückfragen stellen

Beispieldurchführung

Erstellen Sie mit dem Lego Digital Designer aus Einzelbausteinen eine Legofigur mit den folgenden Eigenschaften:
- Die Figur hat eine blaue Mütze auf
- Die Figur hat keine Haare
- Die Figur hat eine Brille im Gesicht
- Die Figur hat ein blaues Jacket an
- Die Figur hat eine blaue Hose an
- Die linke Hand der Figur ist waagerecht gedreht

Testdurchführung jotform Formulargenerator

Aufgabenstellung

Erstellen sie ein Kontaktformular mit dem sich ein Kunde zu einer von mehreren Veranstaltungen ihres Unternehmens anmelden kann. Eine Veranstaltung findet zu mehreren unterschiedlichen Terminen statt.

Das Kontaktformular soll die folgende Funktionalität bieten:

- zwei Radio-Buttons für die Anrede (Pflichtfeld), wobei der Wert „Herr“ vorausgewählt ist
- beschriftete Eingabefelder für die folgenden Informationen:
  - Name (Pflichtfeld)
  - Vorname (optionale Angabe)
  - E-Mail-Adresse (Pflichtfeld, nur gültige E-Mail-Adressen der Form xxx@yyy.zzz erlaubt)
- ein mehrzeiliges Eingabetextfeld für freien Nachrichtentext (optionale Angabe)
- Drop-down Auswahlboxen:
  - mehrere angebotene Veranstaltungen, aus der der Kunde die eine Veranstaltung auswählen kann zu der er sich anmelden möchte (Pflichtfeld)
  - wählt er eine Veranstaltung aus, so sollen in einer weiteren Auswahlbox nur die Termine für diese Veranstaltung angezeigt werden. (Pflichtfeld)
- Ein Kunde soll ein unvollständig ausgefülltes Formular nicht abschicken können.
- Der Kunde soll, nachdem er seine Anmeldung vollständig ausgefüllt und abgeschickt hat, eine Bestätigungsemail mit den Daten der Veranstaltung, zu der er sich angemeldet hat, bekommen.
- Die Daten aus dem Formular sollen per E-Mail an zwei unterschiedliche Empfänger aus ihrem Unternehmen gesendet werden.

durchzuführende Tasks **jotform Formulargenerator**

Task 1		**Zeit. 3 Min**
Beschreibung	Loggen Sie sich auf der Seite mit dem Benutzernamen js_dpl ein. Orientieren Sie sich auf der Oberfläche der Anwendung und verschaffen Sie sich einen Überblick über die Funktionen.	
O.K. Kriterium	Der Benutzer ist eingeloggt, hat einen Überblick über die Organisation der Benutzeroberfläche und die Funktionen der Anwendung	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 2		**neues Formular**
Beschreibung	Erstellen Sie ein neues leeres Formular mit dem Default Style.	
O.K. Kriterium	Der Benutzer hat ein neues (leeres) Formular im Default Style angelegt.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 3	Anrede	
Beschreibung	Fügen Sie dem Formular für die Anrede zwei beschriftete Radio-Buttons hinzu. Dabei soll der Wert „Herr“ die Vorauswahl sein. Die Angabe der Anrede ist ein Pflichtfeld und als solches gekennzeichnet.	
O.K. Kriterium	Das Formular enthält zwei Radio-Buttons zur Auswahl der Anrede, wobei der Button mit dem Wert „Herr“ vorselektiert ist.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 4	Name und Vorname	
Beschreibung	Fügen Sie dem Formular zwei beschriftete einzeilige Texteingabefelder für Name und Vorname hinzu. Das Feld für den Namen soll ein Pflichtfeld und als solches gekennzeichnet sein. Die Texteingabefelder sollen eine Breite von 30 Zeichen haben und die maximale Anzahl einzugebender Zeichen ist ebenfalls auf 30 Zeichen beschränkt.	
O.K. Kriterium	Das Formular enthält zwei 30 Zeichen breite Texteingabefelder jeweils für den Namen und den Vornamen des Kunden. Das Feld zur Eingabe des Namens ist als Pflichfeld gekennzeichnet.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 5		E-Mail Adresse
Beschreibung	Fügen Sie dem Formular ein beschriftetes einzeiliges Texteingabefeld für die E-Mail Adresse hinzu. Das Feld soll ein Pflichtfeld und als solches gekennzeichnet sein. Das Texteingabefeld soll eine Breite von 30 Zeichen (auch maximale Anzahl Zeichen) haben. Legen Sie fest, dass vor dem Abschicken des Formulars die eingegebene E-Mail Adresse auf eine gültige Form überprüft wird.	
O.K. Kriterium	Das Formular enthält ein 30 Zeichen breites Texteingabefeld für die E-Mail Adresse des Kunden. Das Feld ist als Pflichfeld gekennzeichnet. Das Format der E-Mail Adresse wird vor dem Abschicken des Formulars auf Gültigkeit geprüft.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 6		optionale Mitteilung
Beschreibung	Fügen Sie dem Formular ein mehrzeiliges Eingabetextfeld für die optionale Mitteilung hinzu. Das Texteingabefeld soll eine Breite von 30 Zeichen und eine Höhe von 10 Zeichen haben. Das Feld ist kein Pflichtfeld. Es soll mit dem folgenden Text vorbelegt sein: „Hier können Sie uns eine Nachricht hinterlassen.“	
O.K. Kriterium	Das Formular enthält ein 30 Zeichen breites und 10 Zeichen hohes mehrzeiliges Texteingabefeld für eine optionale Mitteilung. Das Feld ist mit dem folgenden Text vorbelegt: „Hier können Sie uns eine Nachricht hinterlassen.“	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 7	Veranstaltungstitel	
Beschreibung	Fügen Sie dem Formular die beschriftete Drop-Down Auswahlbox für die Veranstaltungstitel hinzu. Tragen sie drei Werte für Veranstltungstitel ein. Es soll nur ein Veranstaltungstitel auswählbar sein. Die Auswahl eines Veranstaltungstitels ist ein Pflichtfeld.	
O.K. Kriterium	Das Formular enthält eine Drop-Down Auswahlbox für die Veranstaltungstitel. Die Drop-Down Box ist beschriftet und als Pflichtfeld gekennzeichnet. Es ist nur ein Veranstaltungstitel auswählbar.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 8	Veranstaltungstermine	
Beschreibung	Fügen Sie dem Formular die beschriftete Drop-Down Auswahlbox für die Veranstaltungstermine hinzu. Ein Veranstaltungstitel hat mehrere Termine. Es sollen nur die zu einem Veranstaltungstitel gehörigen Termine auswählbar sein. Die Auswahl eines Veranstaltungstermins ist ein Pflichtfeld.	
O.K. Kriterium	Das Formular enthält eine Drop-Down Auswahlbox für die Veranstaltungstermine. Es sind nur die Termine zu einem ausgewählten Veranstaltungstitel auswählbar. Die Drop-Down Box ist beschriftet und als Pflichtfeld gekennzeichnet. Es ist nur ein Termin auswählbar.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 9	Button zum Abschicken des Formulars	
Beschreibung	Beschriften Sie den Button zum Abschicken des Formulars mit „Anmelden“.	
O.K. Kriterium	Das Formular enthält einen mit „Anmelden“ beschrifteten Button zum Abschicken des Formulars.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 10	Reminder	
Beschreibung	Legen Sie fest, dass ein ausgefülltes Formular an zwei E-Mail Empfänger (eins@infopark.de und zwei@infopark.de) aus Ihrem Unternehmen geht.	
O.K. Kriterium	Der Proband hat festgestellt, dass das Formular nur an einen Empfänger versendet werden kann.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 11	**Bestätigungsemail**	
Beschreibung	Legen Sie fest, dass der Kunde nach Abschicken des Formulars eine Bestätigungsemail mit seinen im Formular angegebenen Daten erhält.	
O.K. Kriterium	Nach Abschicken des Formulars wird eine Bestätigungsemail an den Kunden versandt.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Interview jotform Formulargenerator

Als Einstieg:
- Was sagen Sie spontan zu dem Produkt?
- Was sind Ihrer Meinung nach die Stärken und Schwächen des Produkts

Weitere Fragen/Themen die ggf. durch die Einstiegsfragen nicht betrachtet wurden
- Nennen Sie bitte jeweils ein paar Dinge, die ihnen an dem Programm gut/weniger gut gefallen haben
- Beschreiben Sie bitte ihre Schwierigkeiten bei der Bedienung des Programms während der Aufgabendurchführung.
- Wie schätzen Sie die Verständlichkeit des Programms ein? An welchen Stellen mangelt es Ihres Erachtens nach an Verständlichkeit?
- Was halten Sie hinsichtlich eines ansprechenden Designs für verbesserungswürdig?
- Nennen Sie bitte jeweils ein paar Elemente bei der Bedienung/Navigation, die ihnen eher gut/eher weniger gut gefallen haben
- In welchen Momenten der Bearbeitung haben Sie Freude und positive Anspannung erlebt?
- Welche Funktionen würden Sie sich von dem betrachteten Produkt für den Einsatz in Ihrer Arbeitsdomäne wünschen?

Testdurchführung: formdesk

Aufgabenstellung

Erstellen Sie ein Online- Formular mit dem ein Kunde Probleme mit seinem Informationssystem bei Ihnen melden kann.

- Das Formular enthält beschriftete einzeilige Textfelder zur Eingabe des Namens und der E-Mail Adresse. Beide Felder sind Pflichtfelder und sind als solche gekennzeichnet.
- Das Eingabefeld für die E-Mail Adresse akzeptiert nur E-Mail Adressen in einem gültigen Format.

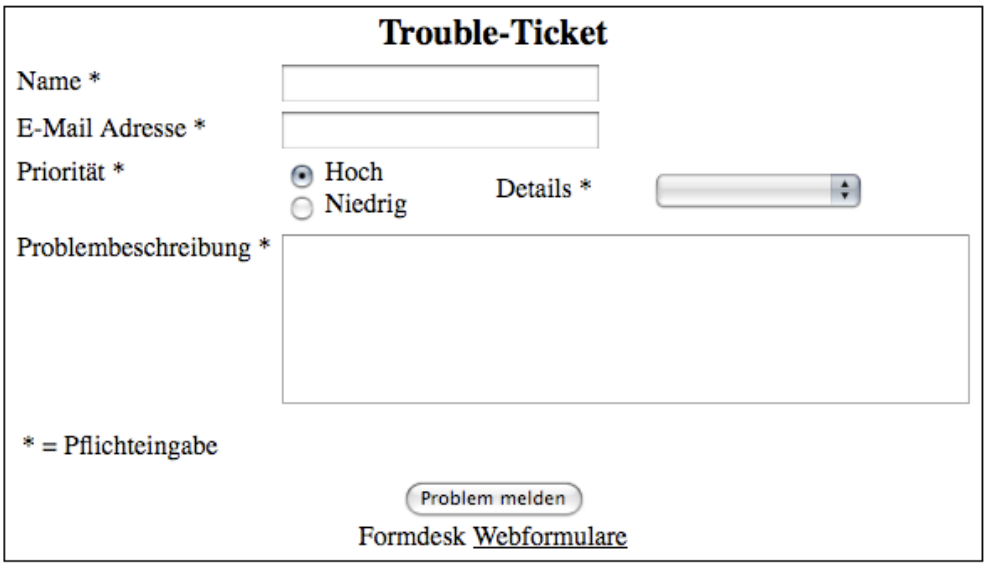
- Der Kunde muss für sein Problem eine Priorität vergeben.
- Setzt der Kunde die Priorität seiner Problemmeldung auf „Hoch“, erscheint eine Auswahlbox in der er Details zu seinem Problem angeben muss.
- Ist die Priorität auf „Niedrig“, so ist die Auswahlbox für die Details nicht zu sehen.

- Das Formular enthält ein beschriftetes mehrzeiliges Texteingabefeld, in dem der Kunde seine Problembeschreibung eingeben muss. Das mehrzeilige Textfeld für die Problembeschreibung ist ein Pflichtfeld und als solches gekennzeichnet.

- Das Formular enthält einen Button zum Versenden des Formulars, welcher mit „Problem melden“ beschriftet ist.

- In Abhängigkeit von der gewählten Priorität wird das Formular an unterschiedliche Empfänger verschickt.

- Der Absender des Formulars bekommt nach dem Abschicken des Formulars eine E-Mail Benachrichtigung, in der er mit Anrede und Namen begrüßt wird und die den Text „wir haben ihre Problemmeldung erhalten“ enthält.



durchzuführende Tasks

Task 1	**Überblick BPMN**	
Beschreibung	Verschaffen Sie sich anhand der auf dem Desktop liegenden Datei bpmn_introduction.pdf einen Überblick über die Business Process Modeling Notation (BPMN) und deren Elemente zur Prozessmodellierung.	
O.K. Kriterium	Der Proband hat einen Überblick über die Elemente der BPMN	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 2	**Überblick BPMN**	
Beschreibung	Überlegen Sie wie der Workflow des Formulars und der entsprechenden E-Mailbenachrichtigungen ist und welche Akteure daran beteiligt sind. Versuchen Sie dieses in BPMN auf einem Blatt Papier oder in dem Programm OmniGraffle zu modellieren.	
O.K. Kriterium	Den Workflow der Aufgabenstellung in BPMN modelliert.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 3	Überblick formdesk (max. 7 Min)	
Beschreibung	Verschaffen Sie sich einen Überblick über die Möglichkeiten der Anwendung formdesk, insbesondere über die Möglichkeiten von Formularen, Workflows, E-Mail-Berichten und Workflow-Berichten.	
O.K. Kriterium	Der Proband hat einen Überblick über die Oberfläche der Anwendung, deren Funktionen sowie Funktionsweise von workflows, Workflow-Berichten und E-Mail-Berichten.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 4	neues Formular	
Beschreibung	Loggen sie sich mit dem Benutzernamen julia.schenk@infopark.de ein und erstellen Sie ein neues Formular mit zentriertem Layout.	
O.K. Kriterium	Der Proband hat ein neues Formular erstellt, welches ein zentriertes Layout hat.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 5	**Überschrift**	
Beschreibung	Fügen Sie dem Formular die Überschrift „Trouble-Ticket“ hinzu.	
O.K. Kriterium	Das Formular hat die Überschrift „Trouble Ticket“	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 6	**Name und E-Mail**	
Beschreibung	Fügen Sie dem Formular beschriftete einzeilige Texteingabefelder zur Eingabe des Namens und der E-Mail Adresse hinzu. Beide Eingabefelder sind Pflichtfelder. Die Felder haben eine Breite von 25 Zeichen und es sind auch maximal 25 Zeichen als Eingabe erlaubt. Das Eingabefeld für die E-Mail Adresse erlaubt nur E-Mail Adressen in einem gültigen Format.	
O.K. Kriterium	Das Formular enthält zwei beschriftete und als Pflichtfelder gekennzeichnete einzeilige Texteingabefelder für den Namen und die E-Mail Adresse, wobei nur E-Mail Adressen in einem gültigen Format akzeptiert werden.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 7	Prioritäten	
Beschreibung	Erstellen Sie zwei beschriftete Radio-Buttons zum Festelgen der Priorität „Hoch“ oder „Niedrig“. Die Angabe einer Priorität ist ein Pflichtfeld und als solches gekennzeichnet. Es darf nur einer der Radio-Buttons auswählbar sein.	
O.K. Kriterium	Das Formular enthält zwei Radio-Buttons zur Vergabe der Priorität „Hoch“ oder „Niedrig“. Die Vergabe einer Priorität ist als Pflichtfeld gekennzeichnet, es kann nur eine Priorität ausgewählt werden.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 8	Details	
Beschreibung	Fügen Sie neben die Radio-Buttons zur Prioritätsvergabe eine mit „Details“ beschriftete Drop-Down Box ein, welche die Werte „technisches Problem“ und „Softwarefehler“ enthält. Es ist nur einer der beiden Werte auswählbar. Legen Sie fest, dass die Auswahlbox für die Details nur dann erscheint, wenn die Priorität „Hoch“ ausgewählt ist. Die Angabe von Details ist bei ausgewählter Priorität „Hoch“ ein Pflichtfeld.	
O.K. Kriterium	Das Formular enthält eine Drop-Down Auswahlbox, welche nur bei ausgewählter Priorität „Hoch“ erscheint. Die Drop-Down Box für die Details ist ein Pflichtfeld und als solches gekennzeichnet. Es ist nur ein Wert aus der Drop-Down Box auswählbar.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 9	Button zum Absenden	
Beschreibung	Nun sind alle Felder des Formulars erstellt. Legen Sie jetzt fest, dass der Button zum Abschicken des Formulars mit „Problem melden“ beschriftet ist.	
O.K. Kriterium	Das Formular enthält einen Button zum Absenden des Formulars, welcher mit „Problem melden“ beschriftet ist.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 10	Benachrichtigung	
Beschreibung	Legen Sie fest, dass der Absender des Formulars nach dem Abschicken des Formulars eine E-Mail Benachrichtigung bekommt, in der er mit seinem Namen begrüßt wird und die den Text „wir haben ihre Problemmeldung erhalten“ enthält. Die Benachrichtigungsemail hat den Betreff „Vielen Dank für Ihre Meldung“.	
O.K. Kriterium	Das Formular hat die Funktionalität, dass der Kunde nach Abschicken des Formulars eine Bestätigung über den Eingang seiner Problemmeldung bekommt. Diese Benachritigungsemail enthält eine personalisierte Anrede und den Text „wir haben ihre Problemmeldung erhalten“.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 11	unterschiedliche Empfänger	
Beschreibung	Legen Sie fest, dass das Formular in Abhängigkeit von der gewählten Priorität an unterschiedliche Empfänger geht. Ein Formular mit ausgewählter hoher Priorität geht an hochprio@infopark.de, ein Formular mit niedriger Priorität geht an niedrigprio@infopark.de.	
O.K. Kriterium	Das Formular hat die Funktionalität, dass je nach ausgewählter Priorität das Formular an unterschiedliche Empfänger geht.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Interview formdesk

Als Einstieg
- Was sagen Sie spontan zu dem Produkt?
- Was sind Ihrer Meinung nach die Stärken und Schwächen des Produkts?
- Wie leicht/schwer viel Ihnen des Verständnis und die Modellierung der Aufgabenstellung in BPMN?

Fragen/Themen die ggf. durch die Einstiegsfragen nicht betrachtet wurden
- Nennen Sie bitte jeweils ein paar Dinge, die ihnen an dem Programm gut/weniger gut gefallen haben.
- Beschreiben Sie bitte ihre Schwierigkeiten bei der Bedienung des Programms während der Aufgabendurchführung.
- Wie schätzen Sie die Verständlichkeit des Programms ein?
- An welchen Stellen mangelt es Ihres Erachtens nach an Verständlichkeit?
- Was halten Sie hinsichtlich eines ansprechenden Designs für verbesserungswürdig?
- Nennen Sie bitte jeweils ein paar Elemente bei der Bedienung/Navigation, die ihnen eher gut/eher weniger gut gefallen haben.
- In welchen Momenten der Bearbeitung haben Sie Freude und positive Anspannung erlebt?
- Welche Funktionen würden Sie sich von dem betrachteten Produkt für den Einsatz in Ihrer Arbeitsdomäne wünschen?

Testdurchführung: WebDSL

Task 1	Überblick WebDSL	
Beschreibung	Verschaffen Sie sich anhand der Dokumente in dem auf dem Desktop liegenden Ordner webdsl einen Überblick über die Konzepte der dömänenspezifischen Sprache WebDSL, welche zur Modellierung von Webanwendungen, also auch Seiten und Formularen, gedacht ist.	
O.K. Kriterium	Der Benutzer hat einen groben Überblick über die Konzepte der WebDSL.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 2	Definition Seite und Formular	
Beschreibung	Versuchen Sie anhand der Dokumentation und der aufgeführten Beispiele in dem Texteditor Textmate in WebDSL eine Seite zu definieren, welche wiederum ein leeres Formular enthält.	
O.K. Kriterium	Der Benutzer hat eine Seite definiert, welche ein leeres Formular enthält.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Task 3	**Definition Formularfelder**	
Beschreibung	Definieren Sie in dem Formular nun zwei Textfelder zur Eingabe eines Namens und einer E-Mail Adresse.	
O.K. Kriterium	Das Formular enthält zwei Definitionen für die Textfelder zur eingabe des Namens und der E-Mail-Adresse.	**erfüllt**
Bemerkungen		
Benötigte Zeit		

Interview WebDSL

Einstieg
- Was sagen Sie spontan zu dieser Art der Problemmodellierung?
- Was meinen Sie könnten die Vor- und Nachteile dieser textuellen Ausdrucksweise in ihrer Arbeitsdomäne sein?
- Wie schätzen Sie die Verständlichkeit der formulierten Beispiele ein?

Anhang D

Screenshots der getesteten Produkte

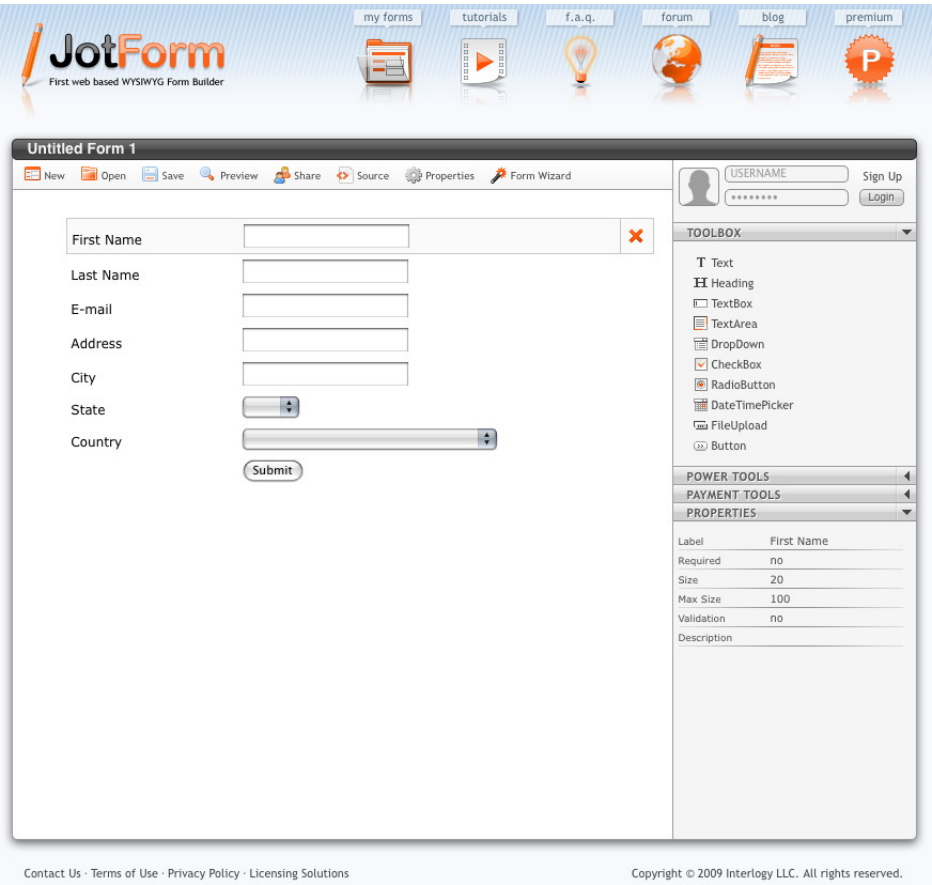


Abbildung D.1: *jotform*: Startseite

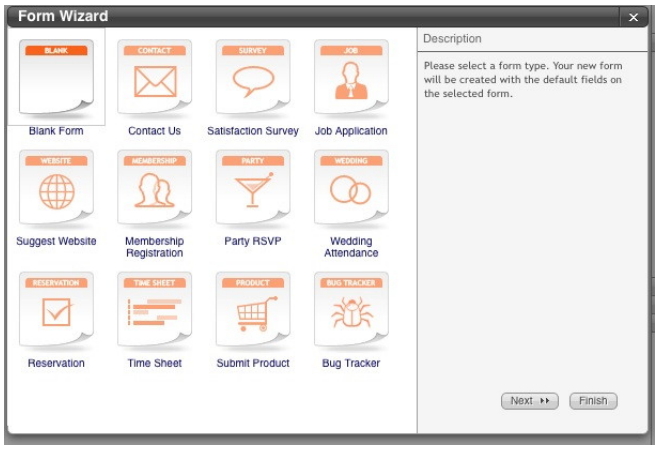


Abbildung D.2: *jotform*: Wizard

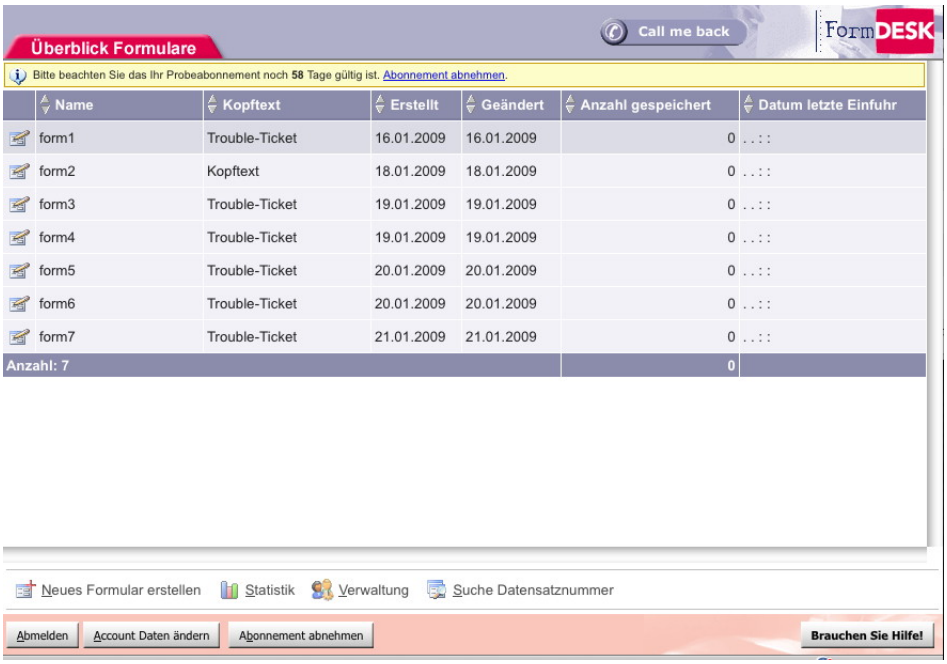


Abbildung D.3: *formdesk*: Übersicht über die Anwendung

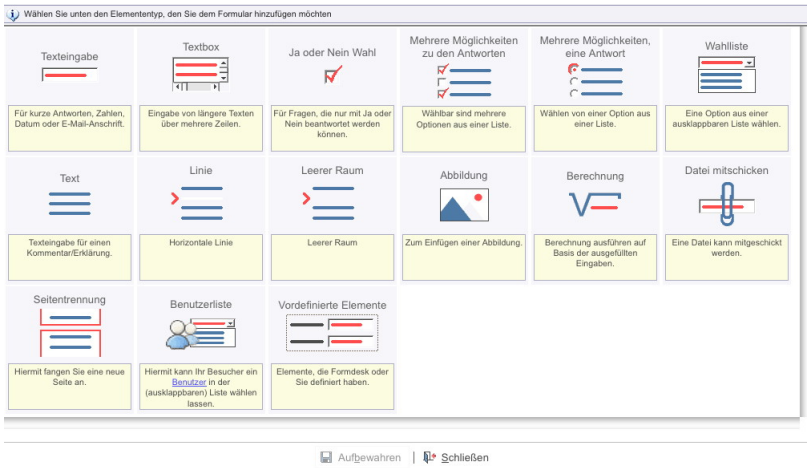


Abbildung D.4: *formdesk*: Auszuwählende Formularelemente

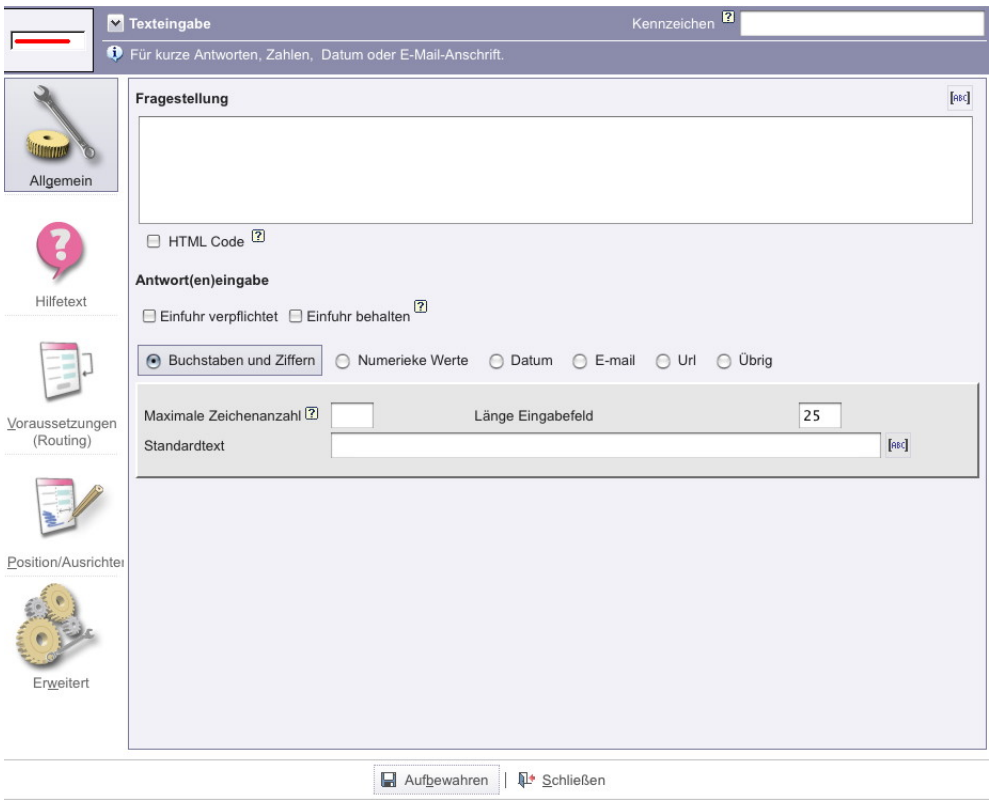


Abbildung D.5: *formdesk*: Bearbeitungsmodus eines Formularelements

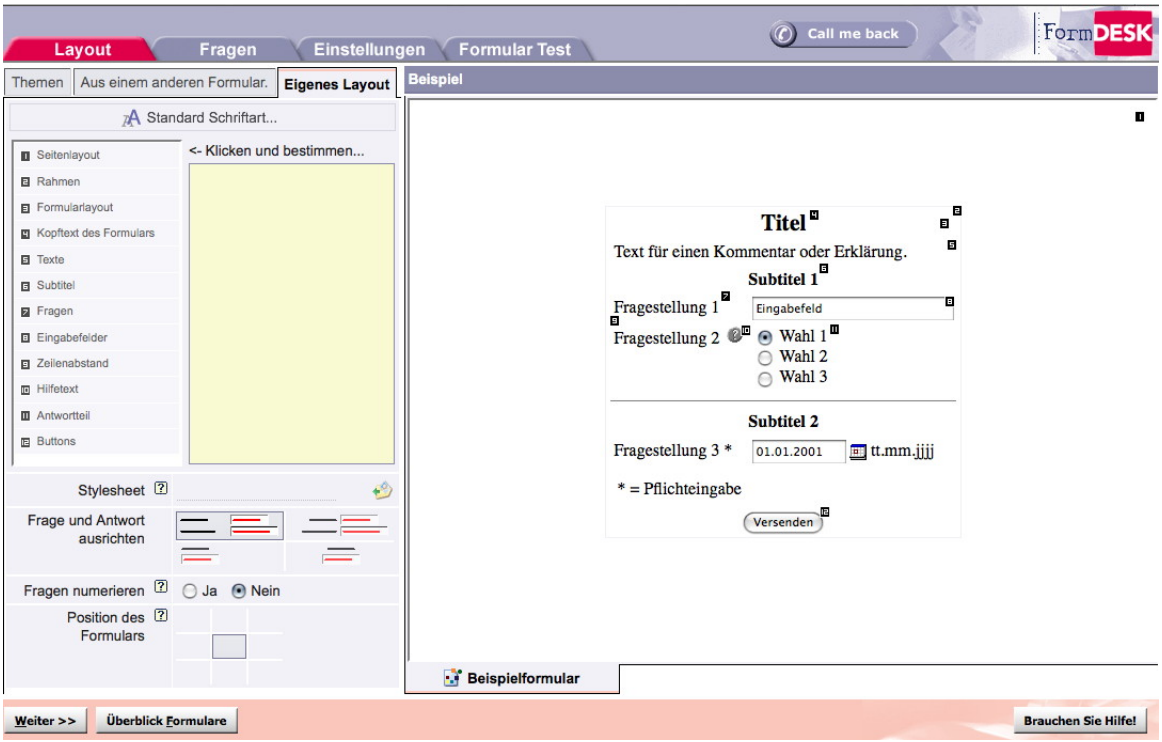


Abbildung D.6: *formdesk*: Schritt der Layoutauswahl

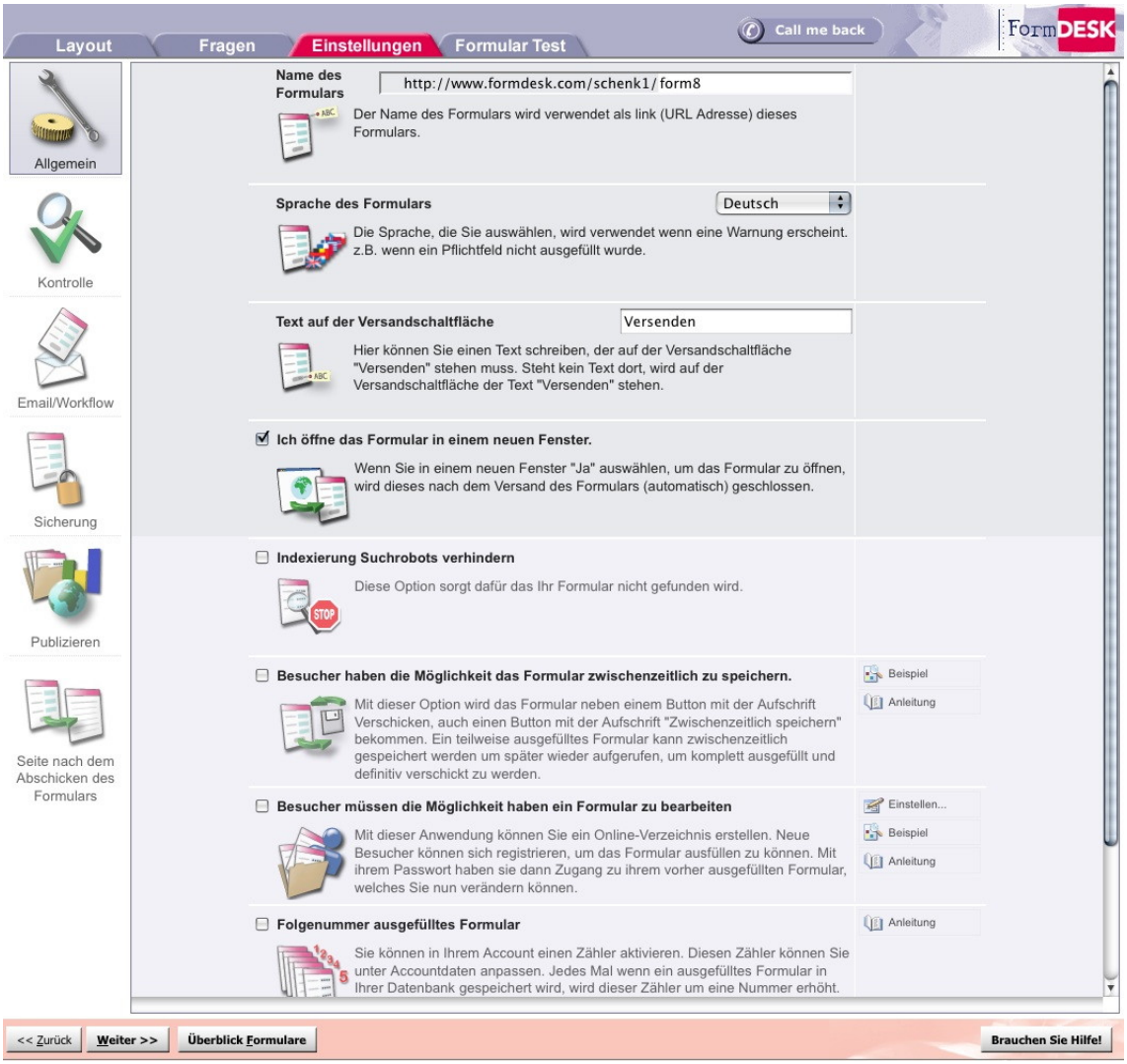


Abbildung D.7: *formdesk*: Allgemeine Formulareinstellungen

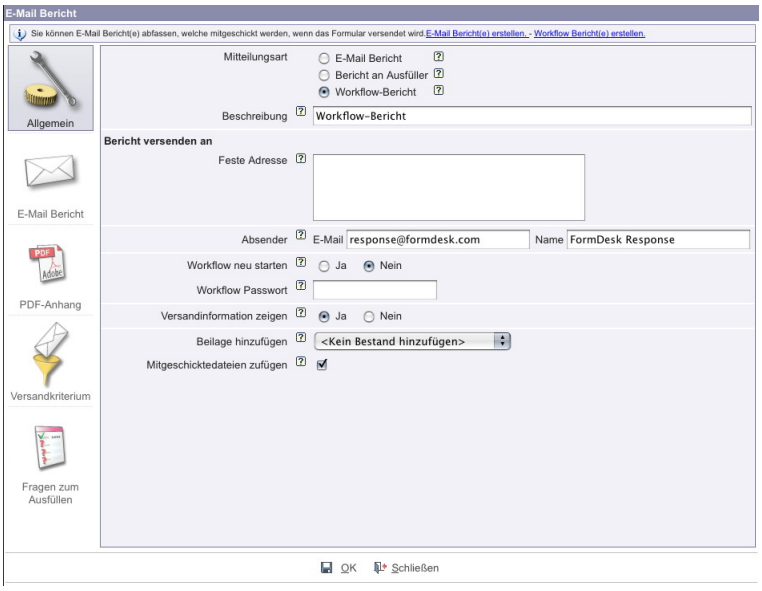


Abbildung D.8: *formdesk*: Unterschiedliche Berichte

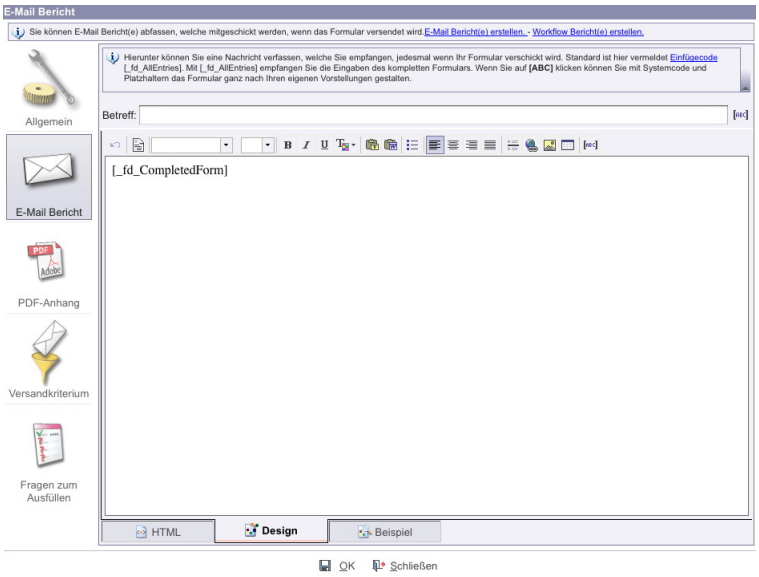


Abbildung D.9: *formdesk*: Bearbeiten des Inhalts eines E-Mail Berichts

Anhang E

Auswertungstabelle der Testdurchführungen

Tabellarische Auswertung der Testdurchführungen mit der Anwendung *jotform*

Anhang E Auswertungstabelle der Testdurchführungen

jotform: tabellarische Auswertung der Testdaten Seite 1

jotform - Auswertung der erhobenen Testdaten

Mikrotask	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung	H.K. Notizen
1	Einloggen in die Anwendung jotform	4	1	- nimmt falsche Maske zum Einloggen - positiv: Feedback über erfolgreiches Einloggen
2	ein neues, leeres Formular erstellen	1	1	- positiv: kommt mit Wizard gut zurecht
3	dem Formular den ‚default' Style zuweisen	1	1	
4	Hinzufügen der Radio-Buttons für die Anrede	1	2	
5	Beschriftung der Radio-Buttons für die Anrede	2	2	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
6	die Werte der Radio-Buttons für die Anrede setzen	4	1	- findet (erkennt) Properties Palette nicht - Hinweis: in rechte Menüspalte schauen
7	Vorauswahl bei den Radio-Buttons auf den Wert „Herr" setzen	1	3	- Versucht ‚selected' über yes/no zu setzen - Hinweis: Wert für ‚selected' angeben - nachdem Wert angegeben, fehlt Feedback darüber was als Vorauswahl gesetzt - anfangs unsicher über Verwendung des vorhandenen Submit Buttons im Formular
8	Kennzeichnung der Anrede als Pflichtfeld	1	1	- positiv: Feedback durch roten Stern
9	Hinzufügen des einzeiligen Textfeldes für den Namen	1	2	- Irritation über D&D Zone
10	Beschriften des einzeiligen Textfeldes für den Namen	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
11	Hinzufügen des einzeiligen Textfeldes für den Vornamen	1	2	- Irritation über D&D Zone
12	Beschriften des einzeiligen Textfeldes für den Vornamen	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
13	Kennzeichnung des einzeiligen Textfeldes für den Namen als Pflichtfeld	1	1	- Kommt problemlos mit dem setzen der Properties der Texteingabefelder zurecht
14	Setzen der Breite und der max. Zeichenanzahl des Textfeldes für den Namen	1	1	
15	Setzen der Breite und der max. Zeichenanzahl des Textfeldes für den Vornamen	1	1	
16	Hinzufügen des einzeiligen Textfeldes für die E-Mail-Adresse	1	2	- Negativer Aspekt über D&D Zone, aber weiß Workaround
17	Beschriften des einzeiligen Textfeldes für die E-Mail-Adresse	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
18	Kennzeichnung des einzeiligen Textfeldes für die E-Mail-Adresse als Pflichtfeld	1	1	
19	Festlegen der Breite und max. Zeichenanzahl für das einzeilige Textfeld für die E-Mail-Adresse	1	1	
20	Validierung der Eingabe für das Textfeld der E-Mail-Adresse festlegen	1	1	
21	Hinzufügen des Textbereiches für die optionale Mitteilung hinzufügen	1	1	- findet richtige Stelle in D&D Zone
22	Beschriftung des Textbereiches für die optionale Mitteilung	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
23	Festlegen der Breite von 30 Zeichen und der Höhe von 10 Zeilen für den Textbereich für die optionale Mitteilung	1	1	
24	Eingabe der Textvorbelegung für den Textbereich für die optionale Mitteilung	1	1	- klickt intuitiv in merz. Textfeld im Formular zur Eingabe der Textvorbelegung
25	Hinzufügen der Drop-Down-Box für die Veranstaltungstitel	1	1	
26	Beschriftung der Drop-Down-Box für die Veranstaltungstitel	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
27	Eintragen der Werte für Drop-Down Auswahl für die Veranstaltungstitel	1	2	- hat beim ersten Mal vergessen den ‚DONE' Button zu klicken

jotform: tabellarische Auswertung der Testdaten Seite 2

Mikrotask	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung	H.K. Notizen
28	Es soll nur ein Wert in der Drop-Down Auswahl für die Veranstaltungstitel auswählbar sein	4	2	- probiert Property „size“ aus, ist sich aber nicht sicher ob dort richtige Einstellung
29	Markierung der Drop-Down Box für die Veranstaltungstitel als Pflichtfeld	1	1	
30	Hinzufügen der Drop-Down-Box für die Veranstaltungstermine	1	1	
31	Beschriften der Drop-Down-Box für Termine	1	1	- beschriftet Element über Doppelklick im Formular, nicht über Label-Property
32	Eintragen der Werte für die Drop-Down Box für die Veranstaltungstitel	1	1	
33	Markierung der Drop-Down Box für die Veranstaltungstitel als Pflichtfeld	1	1	
34	Erkennen, dass die Funktion zum Herstellen der Abhängigkeit zwischen Veranstaltung und Terminen nicht geht	4	4	
35	Den Button zum Abschicken des Formulars mit „Anmelden“ beschriften	1	1	- hat erkannt, dass Submit-Button bereits vorhandens Formularelement ist
36	Herausfinden, dass es nicht möglich ist zwei Empfänger für die Benachrichtigungs-E-Mail mit den Formulardaten einzutragen	3	4	- versucht gesamtes Formular zu aktivieren, um Properties des Formlars zu bekommen - Probiert „Wizard“ Button aus, klickt durch und probiert an richtige Stelle zu kommen - Sieht keine Möglichkeit zweite Empfängeradresse einzutragen
37	Bestätigungs-E-Mail an den Ausfüller des Formulars aktivieren	1	2	- geht wieder über Wizard um an entsprechende Stelle zu kommen - Irritation über Reply-To Feld

jotform: tabellarische Auswertung der Testdaten Seite 3

Mikrotask	H.K. Min/Task	C.D. Orientierung	C.D. Umsetzung	C.D. Bemerkungen	C.D. Min/Task	M.C. Orientierung	M.C. Umsetzung
1	1:24	2	1	- deutsch wäre einfacher - bemängelt „Fachsprache“: z.B. Radio-Button	01:24	1	1
2	0:42	1	1		00:42	1	1
3		4	1	- wollte im Wizard auf ‚finish‘ statt auf ‚next‘ klicken		1	1
4	8:16	1	2	- hätte nicht gewusst was Radio-Buttons sind	03:35	1	4
5		2	1	- liest Hinweis „Klick me to edit“, klickt dort und fügt Beschriftung ein		4	3
6		4	1	- findet Properties-Menü, aber findet nicht die Properties dort zu bearbeiten, will Werte auf per Klick im Formular selbst bearbeiten - Hinweis: bearbeiten der Werte im Properties-Menü		4	1
7		1	4	- Gibt yes bei selected an - Hinweis: Wert angeben für selected - nachdem Wert angegeben, fehlt Feedback darüber was als Vorauswahl gesetzt		1	4
8		1	2	- Sieht nach kurzer Zeit rotes Sternchen als Kennzeichnung für Pflichtfeld, sucht aber Text um rotes Sternchen zu beschreiben.		1	1
9	2:21	1	2	- Probleme mit Drop-Zone und Reihenfolge der Elemente	04:15	2	1
10		1	2	- Beschriftet über Property „Label“ - Übernimmt Beschriftung nicht, beschriftet nochmal im Formularelement direkt		1	1
11		1	2	- negativer Aspekt: Drop Zone		1	2
12		1	2	- negativer Aspekt: versucht erneut über Property „Label“ zu beschriften, funktioniert wieder nicht - beschriftet wieder über Doppelklick im Formularelement selbst - irritiert über wechselnde (flackernde) Ansicht der Propertyleiste, unklar wellche Properties da angezeigt werden (Hinweis: das sind Properties des gesamten Formulars)		1	4
13		1	1			1	1
14		1	1			1	1
15		1	1			1	1
16	1:40	1	2	negativer Aspekt: Drop Zone	01:47	1	2
17		1	2	- beschriftet über Doppelklick im Formularelement selbst, da Label nicht funktioniert		1	2
18		1	1			1	1
19		1	2	- Änderung der Zahl wird nur übernommen, wenn gesamt Zahl markiert ist, Änderung einer einzelnen markierten Ziffer nicht möglich		1	1
20		1	1			1	2
21	1:45	1	2	- negativer Aspekt: Drop- Zone	02:14	1	1
22		1	1			1	1
23		1	1			1	1
24		1	1	- Klickt intuitiv in Textfeldelement um Textvorbelegung einzugeben		1	3
25	3:42	1	1		03:00	1	1
26		1	1			1	1
27		1	1			1	1

jotform: tabellarische Auswertung der Testdaten Seite 4

Mikrotask	H.K. Min/Task	C.D. Orientierung	C.D. Umsetzung	C.D. Bemerkungen	C.D. Min/Task	M.C. Orientierung	M.C. Umsetzung
28		4	2	- probiert aus		4	4
29		1	1			1	1
30	2:53	1	2	- negativer Aspekt: Drop- Zone	02:58	1	1
31		1	1	- beschriftet über Doppelklick im Formularelement selbst		1	1
32		1	2	- irritiert über „flackerndes“ Properties-Menü		1	1
33		1	1			1	1
34		4	4	- nicht umsetzbar		4	4
35	0:50	4	2	- will neuen Button einfügen - Hinweis: es ist schon Submit-Button vorhanden - Probleme Property „Submit Text“ zu bearbeiten: „flackernde“ Properties	01:19	1	1
36	4:20	4	4	- versucht über Anklicken von Submit Button weiter zu kommen - findet Properties des Formulars nicht, probiert im oberen Menü herum - Hinweis: Properties des gesamten Formulars ansehen - sucht Feedback ob/wo zweite E-Mail einzutragen ist - gibt E-Mail Adressen durch Leerzeichen getrennt ein	03:16	4	4
37	2:05	4	4	- sucht wieder in den Properties des Formulars - Hinweis notwendig: Wizard durchklicken - Hinweis: hier ist entsprechende Stelle im Wizard	01:56	4	4

jotform: tabellarische Auswertung der Testdaten Seite 5

Mikrotask	M.C. Bemerkungen	M.C. Min/Task	B.K. Orientierung	B.K. Umsetzung
1		1:41	1	1
2		0:47	1	2
3			1	1
4	- Findet nicht entsprechende Drop-Zone für Radio-Buttons, ist irritiert	4:31	1	2
5	- Versucht über das Submit-Button Element des Formulars ihre Eingaben zu bestätigen - Hinweis: Submit-Button ist ein Formularelement - Werte beschriften ist klar, da in den Properties entsprechende Options-Werte gesetzt waren - inkonsistente Bedienung: Das Wort „Label“ wird im Formularelement aber nicht angezeigt, also unklar was die Property Label ist - beschriftet über Label-Property im Properties Menü		4	1
6	- Weiß nicht wo Werte für die Radio-Buttons zu setzen		3	4
7	- irritiert über yes/no Option - Hinweis: Werte für selected eingeben - nachdem Wert angegeben, fehlt Feedback darüber was als Vorauswahl gesetzt		1	1
8			4	4
9		3:40	2	2
10	- beschriftet über Label-Property im Properties Menü		1	1
11	- negativer Aspekt: Drop-Zone		1	1
12	- beschriftet über Label-Property im Properties Menü - Beschriftung wird nicht übernommen, sucht Möglichkeit zum „Übernehmen“ - Hinweis: Direkt im Formularelement beschriften		1	1
13			1	1
14			1	1
15			1	1
16	- negativer Aspekt: Drop-Zone	3:16	1	1
17	- beschriftet wieder über Label-Property im Properties Menü, funktioniert nicht - beschriftet danach im Formularelement direkt		1	1
18			1	1
19			1	1
20	-negativer Aspekt: hätte gerne nähre Informationen/ Möglichkeiten bzgl. der Validierungsart		1	1
21		2:00	1	1
22			1	1
23			1	1
24	- Irritation über inkonstistent Bedieung: Der Vorbelegungstext ist direkt im Formularelement und nicht in den Properties einzugeben		1	1
25		3:06	1	1
26			1	1
27			1	1

jotform: tabellarische Auswertung der Testdaten Seite 6

Mikrotask	M.C. Bemerkungen	M.C. Min/Task	B.K. Orientierung	B.K. Umsetzung
28	- Es wird nicht klar, ob/wo man einstellen kann, dass nur eine Veranstaltung auswählbar ist - Überlegt, was Property size zu bedeuten hat		4	1
29			1	1
30		3:30	1	1
31			1	1
32			1	1
33			1	1
34	- sucht nach Verküpfungsmöglichkeit der Veranstaltungen und deren Termine		4	4
35		0:30	1	1
36	- sucht Möglichkeit wo E-Mail-Empfänger einzugeben - Hinweis: ist Property des gesamten Formulars - Gibt einfach zweite E-Mail-Adresse durch Komma getrennt ein - negativer Aspekt: fehlendes Feedback zu eingegebenen Daten	2:30	2	4
37	- sucht Möglichkeit für Bestätigungsemail ebenfalls in den Properties des gesamten Formulars - probiert den „Wizard“, schließt diesen aber wieder - Hinweis: muss durch Wizard durchklicken	3:28	4	4

jotform: tabellarische Auswertung der Testdaten Seite 7

Mikrotask	B.K. Bemerkungen	B.K. Min/Task	A.V. Orientierung	A.V. Umsetzung
1		1:34	1	1
2	- im Wizard ist unklar warum ‚next' und ‚finish' möglich		1	2
3			1	1
4	- negativer Aspekt: Drop-Zone	1:21	1	3
5	- Irritiert über „Click me to edit", weiß nicht wo Beschriftung zu setzen		3	4
6	- Problem: flackernder Properties-Inspektor - klickt Option 2 im Formularelement an, geht dann auf Löschen, gesamtes Element wird gelöscht - weiß nicht wie 3. Option zu löschen, Hinweis notwendig		2	1
7	- nachdem Wert angegeben, fehlt Feedback darüber was als Vorauswahl gesetzt		2	4
8	- fügt zunächst eigene Kennzeichnung an Beschriftung hinzu - nach Hinweis benutzt er die Property „required" - Bestätigt Eingaben über den Submit-Button des Formulars		1	1
9	- negativer Aspekt: Drop-Zone	6:03	1	1
10	- Beschriftet direkt im Formularelement		1	1
11	- vermisst Copy-Paste Funktion für Textfelder einfügen			2
12			1	1
13			1	1
14			1	1
15			1	1
16		3:50	1	1
17			1	2
18			1	3
19			1	3
20			1	3
21		0:54	1	1
22			1	1
23			1	2
24	- findet sofort Möglichkeit für Eingabe der Textvorbelegung, klickt intuitiv in das Feld des Formularelements		1	1
25		1:01	1	1
26			1	1
27			1	4

jotform: tabellarische Auswertung der Testdaten Seite 8

Mikrotask	B.K. Bemerkungen	B.K. Min/Task	A.V. Orientierung	A.V. Umsetzung
28	- Unklarheit über Property „Special“		4	2
29			1	1
30		2:10	1	1
31			1	1
32			1	4
33			1	1
34	- Sucht Möglichkeit um Verknüpfung herzustellen, findet sie nicht - Hinweis: nicht umsetzbar		3	3
35			1	1
36	- Schaut in den Formularproperties nach, gibt dort durch Komma getrennt die zwei E-Mail-Adressen ein - vermisst Feedback über Eingaben, würde gerne testen	0:33	2	4
37	- Sucht in den Formularproperties, probiert dort Optionen herum - verwendet falsche Option - vermisst erklärenden Text	0:38	4	4

jotform: tabellarische Auswertung der Testdaten Seite 9

Mikrotask	A.V. Bemerkungen	A.V. Min/Task	Mittelwert Problemgrad
1		3:00	1,4
2	- hat nicht erwartet, dass sich ein Wizard öffnet	1:08	1,2
3			1,3
4	- hat Probleme mit der Drop-Zone - fügt Element zufällig durch Doppelklick ein	6:04	1,8
5	- versucht es über die Property „Label“, funktioniert nicht - versucht ein TextBox Element als Beschriftung hinzuzufügen - Hinweis: Bug, muss direkt im Formularelement Beschriftung vornehmen - negativer Aspekt: keine Markierung wie Rahmen etc. wo die Beschriftung im Formularelement zu setzen ist		2,6
6			2,5
7	- kommt nicht zurecht mit Property „Selected“ - kann Property zunächst nicht anklicken - irritiert über flackernde Properties - kann Vorauswahl nicht setzen, trotz Hinweis, dass ein Wert gesetzt werden muss - nachdem Wert angegeben, fehlt Feedback darüber was als Vorauswahl gesetzt		2,2
8			1,7
9		3:46	1,5
10			1,1
11	- negativer Aspekt: Drop-Zone, Reihenfolge der Elemente - hat TextArea hinzugefügt: falsch - hat Probleme mit flackernden Properties		1,4
12			1,4
13			1,0
14			1,0
15			1,0
16		1:59	1,3
17	- irritiert über auf- und zuklappenden Properties-Inspektor		1,3
18			1,2
19	- extreme Probleme Property Wert anzuklicken/zu markieren		1,3
20			1,3
21		1:16	1,1
22			1,0
23	- extreme Probleme Property Wert anzuklicken/zu markieren		1,1
24			1,2
25		5:51	1,0
26			1,0
27	- extreme Probleme Property Inspektor für Formularelement anzuzeigen und darin Werte zu bearbeiten - Proband kommt nicht weiter - Hilfe des Testadministrators notwendig: Option anklicken		1,4

jotform: tabellarische Auswertung der Testdaten Seite 10

Mikrotask	A.V. Bemerkungen	A.V. Min/Task	Mittelwert Problemgrad
28	- weiß nichts mit Property size anzufangen		3,1
29			1,0
30		4:45	1,1
31			1,0
32	- extreme Probleme Property Inspektor für Formularelement anzuzeigen und darin Werte zu bearbeiten - Proband kommt nicht weiter - Hilfe des Testadministrators notwendig: Option anklicken - wünscht sich, dass eingegebene Werte als Datumswerte erkannt werden		1,4
33			1,0
34			3,8
35		0:26	1,4
36	- trägt die E-Mail-Adressen durch Komma getrennt ein bemängelt, dass nicht komplett zu sehen was eingetragen wurde	2:27	3,5
37		4:41	3,5

Tabellarische Auswertung der Testdurchführungen mit der Anwendung *formdesk*

formdesk: tabellarische Auswertung der Testdaten Seite 1

formdesk - Auswertung der erhobenen Testdaten

Mikrotask	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung
1	groben Überblick über die BPMN verschaffen	1	1
2	Extrahieren und Modellieren des Workflows in BPMN (oder ähnlich)	2	3
3	Überblick über die Anwendung formdesk verschaffen	1	1
4	Einloggen in die Anwendung formdesk	1	1
5	ein neues Formular erstellen	1	2
6	ein zentriertes Layout für das neue Formular setzen	1	4
7	dem Formular die Überschrift „Trouble-Ticket“ hinzufügen	1	1
8	Hinzufügen des einzeiligen Texteingabefeldes für den Namen	1	1
9	Beschriften des einzeiligen Textfeldes für den Namen	1	2
10	Setzen der Breite und max. Zeichenanzahl (25) für das Textfeld für den Namen	1	1
11	Texteingabefeld für den Namen als Pflichtfeld setzen	1	2
12	Hinzufügen des einzeiligen Texteingabefeldes für die E-Mail-Adresse	1	1
13	Beschriften des einzeiligen Texteingabefeldes für die E-Mail Adresse	1	1
14	Setzen der Länge für das Texteingabefeld für die E-Mail Adresse	1	1
15	Texteingabefeld für die E-Mail-Adresse als Pflichtfeld setzen	1	1
16	Validierung der E-Mail-Adresse sicherstellen	2	2
17	Hinzufügen der Radio Buttons für die Priorität	1	1
18	Beschriftung der Radio-Buttons für die Priorität	1	1
19	Setzen der zwei Werte der Radio-Buttons für die Priorität	1	2
20	Angabe der Priorität als Pflichtfeld setzen	1	1
21	Hinzufügen der Drop-Down-Box für die Details	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 2

Mikrotask	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung
22	Positionierung der Drop-Down-Box für die Deatails neben die Drop-Down-Box für die Priorität	1	2
23	Beschriftung der Drop-Down Box für die Details	1	1
24	Optionswerte der Drop-Down Box für die Details setzen	1	1
25	Nur einer der Werte darf auswählbar sein	1	1
26	Angabe der Details ist Pflichtfeld	1	1
27	Einstellen, dass die Drop-Down Box für Details nur bei Auswahl des Wertes „hoch“ in der Drop-Down-Box für die Priorität erscheint	4	4
28	Beschriften des Buttons zum Abschicken des Formulars mit „Problem melden“	3	2
29	Bestätigungs-E-Mail an den Ausfüller des Formulars aktivieren	3	4
30	Betreff der Bestätigungs-E-Mail an den Ausfüller setzen	4	1
31	Text in der Bestätigungs-E-Mail an den Ausfüller setzen	1	1
32	Personalisierte Anrede in der Bestätigung-E-Mail an den Ausfüller setzen	4	3
33	Workflowbericht erstellen, der bei gewählter Priorität „hoch“ an einen Empfänger aus dem Unternehmen geht	1	1
34	Empfängeradresse für den Workflowbericht bei ausgewählter Priorität „hoch“ setzen	1	4
35	Einstellen, dass der Workflowbericht auch nur bei ausgewählter Priorität „hoch“ verschickt wird	2	1
36	Workflowbericht erstellen, der bei gewählter Priorität „niedrig“ an einen Empfänger aus dem Unternehmen geht	1	1
37	Empfängeradresse für den Workflowbericht bei ausgewählter Priorität „niedrig“ setzen	1	1
38	Workflowbericht erstellen, der bei gewählter Priorität „niedrig“ an einen Empfänger aus dem Unternehmen geht	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 3

Mikrotask	H.K. Bemerkungen	H.K. Min/Task	C.D Orientierung	C.D. Umsetzung
1	- schaut sich Konzepte de BPMN selbständig in pdf an	6:16	4	4
2	- modelliert in OmniGraffle, kennt den Umgang mit diesem Programm - schaut immer wieder in BPMN Referenz nach - sieht Problem, dass die Aufgabenstellung Informationen darüber enthält, was das Formular enthält, er aber einen Ablauf modellieren soll - erkennt Schwierigkeit der Transferleistung	16:00	4	4
3		3:43	1	1
4		1:54	1	1
5	- findet schnell Menüpunkt (unten) um neues Formular zu erstellen		1	1
6	- sieht eine Formularvorschau - Hinweis: auf „weiter“ klicken um Layout zu übernehmen		1	1
7	- macht Dopelklick auf „Kopftext“	1:00	1	2
8		4:37	2	1
9	- negativer Aspekt: Bezeichnung „Fragestellung“ für Titel des Feldes irritiert - weitere Optionen sind verwirrend		1	2
10			1	1
11	- Bezeichnung „Einfuhr verpflichtet“ für Pflichtfeld verwirrt		1	1
12			1	1
13			1	1
14			1	1
15			1	1
16	- hat zunächst vergessen Validierung einzustellen, sucht im oberen Menü die Möglichkeit in Bearbeitungsmodus des Feldes zurück zu kommen - sieht dort die Option „E-Mail“, vermutet, dass es sich hierbei um die Validierung handelt.		1	1
17		1:35	1	4
18			1	1
19	- liest Hinweistext des Programms um zu wissen in welcher Form die Optionswerte einzugeben sind		4	3
20			1	1
21		9:08	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 4

Mikrotask	H.K. Bemerkungen	H.K. Min/Task	C.D Orientierung	C.D. Umsetzung
22	- findet im linken Menü den Punkt „Ausrichten“ - gewünschter Auswahlpunkt ist nicht auswählbar, erst nachdem voranstehender Menüpunkt aktiviert wurde --> unklar warum			1
23			1	1
24			1	1
25			1	1
26			1	1
27	- Hinweis: obere Registerkarten repräsentieren einzelne Schritte - Hinweis: Abhängigkeit ist Eigenschaft des Felds - Hinweis: Es ist eine Voraussetzung für dieses Feld, daraufhin orientiert sich Benutzer im linken Menü - Probleme bei Begriffen zur Bedingungsdefinition, Hilfestellung notwendig - Lösch-Icon als solches nicht erkennbar		3	4
28	- sucht Button in Liste der Elemente die hinzugefügt werden können, findet ihn dort nicht, klickt auf ‚weiter‘		2	4
29	- sucht unter Einstellungen Möglichkeit Bestätigungsemail einzustellen - schaut im linken Menü, findet dort die Option „Email/ Workflow“ , findet sich innerhalb diese Menüs aber nicht zurecht - Hinweis: Unterschied Workflowbericht (intern) und E-Mail Bericht (an Kunde)	9:24	4	4
30	- Bei Berichterstellung „Allgemein“: Beschreibung=Betreff?, sucht Stelle wo Inhalt des Berichts zu beschreiben ist - Hinweis: im Linken Menü schauen für Betreff etc. setzen (Menüpunkt „E-Mail Bericht“)		4	1
31			1	1
32	- erkennt Möglichkeit der Variablensubstitution - braucht Hinweis zum Einfügen von Variablen, erkennt entsprechendes Icon im Menü nicht		4	3
33			1	1
34	- Bei „Bericht versenden an“ aktiviert er die Option „E-Mail“ und gibt zusätzlich eine feste Adresse an --> falsch		4	1
35	- findet im Linken Menü „Versandkriterium“ und erstellt problemlos Bedingung		3	1
36			1	1
37			1	1
38			1	1

formdesk: tabellarische Auswertung der Testdaten Seite 5

Mikrotask	C.D. Bemerkungen	C.D. Min/Task	M.C. Orientierung	M.C. Umsetzung
1			4	4
2			4	4
3		1:40	1	1
4		1:03	1	1
5			2	1
6	- erkennt, dass offentsichtlich schon zentriertes Layout ausgewählt und klickt auf „Weiter“		4	4
7	- macht intuitiv Doppelklick auf „Kopftext“ - probiert Wert durch „Enter“ drücken zu übernehmen - irritiert über Bezeichnung „Aufbewahren“	0:51	4	1
8	- geht über oberes Menü	2:46	4	2
9	- negativer Aspekt: irritiert über Bezeichnung „Fragestellung“		1	2
10			1	1
11			1	2
12	- geht über Button „Element hinzufügen“		1	1
13			1	1
14			1	1
15			1	1
16			1	1
17	- geht über Button „Element hinzufügen“ - findet nicht richte Bezeichnung für Radio-Buttons - Hinweis: „mehrere Möglichkeiten, eine Antwort“ Element	3:29	2	1
18			1	1
19	- Findet nicht Stelle wo Werte einzugeben sind - Hinweis: Werte in Feld eingeben, - liest danach Hinweis vom Programm - probiert Option „alternative Wahl“, irritiert über Bedeutung		2	2
20			1	1
21		5:38	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 6

Mikrotask	C.D. Bemerkungen	C.D. Min/Task	M.C. Orientierung	M.C. Umsetzung
22				1
23			1	1
24			1	1
25				1
26			1	1
27	- versucht diese Bedingung bei den Radio-Buttons für die Priorität einzustellen - sucht in den Optionen des linken Menüs im Drop.Down Box Bearbeitungsmodus - Probleme bei der Formulierung der Bedingung, auch wegen schlechten Wordings - Unklarheit über Bedeutung der Begriffe „verriegeln, erlauben, sichtbar machen, unsichtbar machen - Hinweis: Klärung der einzelnen Begriffe - Feedback über Funktion fehlt		2	1
28	- klickt intuitiv auf ‚weiter', findet jedoch Stelle nicht wo Beschriftung zu ändern - Hinweis: Stelle ist schon richtig - findet daraufhin die Stelle	1:42	3	4
29	- geht auf ‚weiter' --> falsch, findet nicht Stelle um Bericht einzustellen - Hinweis: zurück und im linken Menü schauen unter „E-Mail/Workflow" - Unterschied „E-Mail Bericht" und „Bericht an Ausfüller" unklar - Hinweis: Bericht an Ausfüller auswählen	4:12	2	1
30	- findet nicht Stelle wo Betreff der Bestätigungsemail einzugeben - will Betreff unter Allgemein -> Beschreibung eingeben - Hinweis: im linken Menü unter „E-Mail Bericht"		3	1
31			1	1
32	- findet nicht Möglichkeit variabel den Namen einzufügen - Hinweis: Hinweistext des Programms lesen		4	4
33		3:52	4	4
34	- erkennt nicht Stelle wo Empfängeradresse einzutragen - Hinweis: Bericht soll an feste Adresse gehen		1	1
35			2	4
36			1	1
37			1	1
38			1	1

formdesk: tabellarische Auswertung der Testdaten Seite 7

Mikrotask	M.C. Bemerkungen	M.C. Min/Task	B.K. Orientierung	B.K. Umsetzung
1	- findet Modellierungskonstrukte schwierig zu verstehen - empfindet Abneigung gegenüber der Modellierung	5:34	4	4
2		5:20	4	4
3		1:36	1	1
4			1	1
5			2	1
6	- Es besteht Unklarheit darüber, wie das Layout auszuwählen ist und ob das angezeigte Formular ein Beispiellayout oder schon die Vorschau eines Formulars ist - Probiert verschiedenste Menüpunkte, findet sich nicht zurecht - Hinweis notwendig, wie Layouts (Themen) auszuwählen sind - Irritation über inkonsistente Bezeichung Layout/Thema		4	4
7	- versucht in Layoutvorschau Formularelemente anzuklicken - Hinweis: muss durch „weiter“ den Layout Schritt verlassen	1:35	2	2
8	- weiß nicht, ob jetzt durch „weiter“ oder durch „Element hinzufügen“ weiter zu kommen - Hinweis: über Element hinzufügen - negativer Aspekt: auswählbare Elemente öffnen sich in einem neuen Fenster, sind also nicht jederzeit übersichtlich dargestellt	3:23	1	1
9	- leichte Unsicherheit: Fragestellung=Beschriftung?		1	1
10			1	1
11	- negativer Aspekt: Bezeichnung „Einfuhr verpflichtet“		2	1
12			1	1
13			1	1
14			1	1
15			1	1
16			1	2
17		2:25	2	1
18			1	1
19	- liest Hinweistext des Programms, kommt dann gut zurecht - negativer Aspekt: Irritation über „Alternative Wahl“		1	1
20			1	1
21		2:34	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 8

Mikrotask	M.C. Bemerkungen	M.C. Min/Task	B.K. Orientierung	B.K. Umsetzung
22			3	1
23			1	1
24			1	1
25			1	1
26			1	1
27	- schaut im linken Menü und findet Option Voraussetzungen - definiert problemlos Bedingungen		2	2
28	- probiert über „Element hinzufügen“, findet kein Button-Element - probiert über „weiter“, findet nicht Stelle für Beschriftung - Hinweis: Stelle ist schon richtig, Button ist für das Versenden zuständig	2:54	3	2
29	- benutzt den vorhandenen E-Mail Bericht	9:56	2	3
30			3	1
31			1	1
32	- Hinweis: Hinweistext des Programms lesen - kommt nicht zurecht mit Beschreibung der Variablensubstitution - Hinweis notwendig über Verwendung der Variablen - fügt Variable über Icon neben Betreff ein, daher erscheint Variable im Betreff und nicht im Inhalt - Hinweis: Icon in dem Menü für den Inhalt benutzen		2	3
33	- Unklarheit darüber, dass neuer Workflowbericht erstellt werden muss - Hinweis notwendig	3:41	1	4
34			1	1
35	- erstellt in diesem einen Bericht zwei Bedingungen für Versandkriterium - Hinweis: pro Bedingung ein Bericht notwendig		2	1
36			1	1
37			1	1
38			1	1

formdesk: tabellarische Auswertung der Testdaten Seite 9

Mikrotask	B.K. Bemerkungen	B.K. Min/Task	A.V. Orientierung	A.V. Umsetzung
1	- Proband weigert sich, sich mit dem Konzept zu beschäftigen		3	3
2			3	3
3		3:23	1	1
4		1:22	1	1
5	- negativer Aspekt: irritiert über Position des Buttons „Neues Formular erstellen“		2	1
6	- kommt nicht zurecht mit Auswahlart für Layouts		3	4
7	- versucht die Überschrift im Layoutmodus zu bearbeiten - navigiert über obere Registerkarten zu Fragen, erkennt hier richtige Stelle - findet das Konzept zur Bearbeitung des Titels nervig und unschön	1:45	2	4
8	- geht über den Button „Element hinzufügen“	2:46	1	1
9			1	2
10			1	1
11	- Irritiert über Bezeichnung „Einfuhr verpflichtet“		1	2
12			1	1
13			1	1
14			1	1
15			1	1
16	- Feedback fehlt, ob durch die Option „E-Mail“ wirklich eine Validierung erfolgt		1	1
17	- vermisst visuelles Feedback über ausgewähltes Element	1:31	1	1
18			1	1
19			1	2
20			1	1
21	- positiver Aspekt: es besteht die Möglichkeit Elemente zu kopieren und einzufügen	4:10	1	1

formdesk: tabellarische Auswertung der Testdaten Seite 10

Mikrotask	B.K. Bemerkungen	B.K. Min/Task	A.V. Orientierung	A.V. Umsetzung
22			1	1
23			1	1
24			1	2
25			1	1
26			1	1
27	- negativer Aspekt: Bezeichnungen „verriegeln“, etc... - vermisst Feedback der Funktion einblenden und ausblenden		4	3
28	-probiert über „Element hinzufügen“, findet kein Button-Element - probiert über „weiter“, findet die Stelle für Beschriftung - Unklarheit darüber, wo der Button im Formular erscheint	1:06	3	2
29	- Unklarheit darüber an wen der vorhandene E-Mail Bericht geht		4	1
30	- setzt Subject zunächst in Allgemein --> Beschreibung		4	2
31			1	1
32	- erkennt Mechanismus der Formularvariablen - liest Hinweistext des Programms zu den Formularvariablen - klickt auf [abc] Icon neben der Betreffzeile - kopiert sich Variable aus Betreff heraus		3	4
33	- erstellt neuen Bericht - Unklarheit über Art des erstellten Berichts - Hinweis: Workflowbericht	4:26	1	4
34			1	1
35			2	1
36	- Kopiert Bericht von hoher Priorität und bearbeitet diesen		1	1
37			1	1
38			1	1

formdesk: tabellarische Auswertung der Testdaten Seite 11

Mikrotask	A.V. Bemerkungen	A.V. Min/Task	Mittelwert Problemgrad
1			3,2
2		8:16	3,5
3		5:42	1,0
4			1,0
5	- findet Positionierung unten des Buttons nicht vorteilhaft		1,4
6	- klickt herum, hat keinen genauen Überblick wo/wie das Layout einzustellen ist - weiß nicht wie das Layout zu übernehmen		3,0
7	- versucht in der Layout-Ansicht den Titel zu bearbeiten - Hinweis: Layout-Modus durch „weiter“ verlassen - negativer Aspekt: Bearbeitungsoptionen sind nicht klar	3:21	2,0
8	- geht über Button „Element hinzufügen“ im Formular	2:24	1,5
9	- negativer Aspekt: Bezeichnung „Fragestellung“		1,4
10			1,0
11	- negativer Aspekt: Bezeichnung „Einfuhr verpflichtet“		1,4
12			1,0
13			1,0
14			1,0
15			1,0
16			1,3
17	- geht über Button „Element hinzufügen“ im Formular	1:46	1,5
18			1,0
19	- negativer Aspekt: Unklarheit über Option „Alternative Wahl“		1,9
20			1,0
21		7:07	1,0

formdesk: tabellarische Auswertung der Testdaten Seite 12

Mikrotask	A.V. Bemerkungen	A.V. Min/Task	Mittelwert Problemgrad
22			1,4
23			1,0
24	- negativer Aspekt: kein visuelles Feedback darüber, dass er gerade die Optionen einer Drop-Down Box bearbeitet		1,1
25			1,0
26			1,0
27	- Unklarheit über Begriffe „verriegeln“, etc.		2,9
28	- überlegt, ob Button vom System hinzu gefügt wird, oder ob er das selbst machen muss, schaut in Menü der einzufügenden Elemente nach, findet es nicht - klickt auf „weiter“	1:05	2,8
29	- Irritiert über Bezeichnung E-Mail Bericht - Hinweis: E-Mail Bericht ist richtig	7:58	2,8
30	- Setzt Betreff in Allgemein --> Beschreibung - Hinweis: Hilfetext zu Beschreibung lesen - negativer Aspekt: linke Navigation ist schlecht gemacht, nicht strukturiert, leitet den Benutzer nicht		2,4
31			1,0
32	- liest Hinweistext des Programms bzgl. Formularvariablen, findet jedoch Icon nicht - findet [abc] neben Betreff, sieht nicht wo er die Variable eingefügt hat - Hinweis: schau mal in Betreff, es gibt noch ein zweites Icon		3,4
33	- erstellt neuen Bericht, will aber E-Mail Bericht erstellen - klickt Workflowbericht an, erkennt aber nicht Möglichkeit des Versandkriteriums - Hinweis: es muss ein Workflowbericht sein	4:20	2,2
34			1,6
35	- hat allgemein Probleme mit linker Navigation, erkennt aber allmählich Bedienungskonzept		1,9
36			1,0
37			1,0
38			1,0

Tabellarische Auswertung der Testdurchführungen mit der Anwendung *perForm*

Anhang E Auswertungstabelle der Testdurchführungen

perForm: tabellarische Auswertung der Testdaten Seite 1

perForm - Auswertung der erhobenen Testdaten

Task Nr.	Mikrotask Nr.	Beschreibung Mikrotask	H.K. Orientierung	H.K. Umsetzung	H.K. Notizen
1	1	Überblick verschaffen	1	1	
2	2	neues Formular mit default Layout erstellen	1	1	
3	3	Texteingabefeld für den Nachnamen hinzufügen	1	2	- Formularelement Texteingabefeld sollte weiter vorne in der Liste der Formularelemente erscheinen - irritiert über D&D revert Effekt
3	4	Texteingabefeld für den Nachnamen beschriften	2	2	- versucht zunächst im WYSIWYG Formular die Beschriftung zu ändern, geht dann aber in den Properties Inspektor - will Beschriftung in der Property ‚interne Beschreibung' setzen
3	5	Texteingabefeld für den Nachnamen als Pflichtfeld setzen	1	2	- bestätigt die Eingabe der Auswahlliste nicht mit dem OK Button --> negativer Aspekt, dass Eingabe mit Button bestätigt werden muss
3	6	für das Texteingabefeld für den Nachnamen eine Breite von 25 Zeichen einstellen	1	1	
3	7	für das Texteingabefeld für den Nachnamen eine maximale Anzahl Zeichen (25 Zeichen) einstellen	1	1	
4	8	Texteingabefeld für die E-Mail Adresse hinzufügen	1	1	
4	9	Texteingabefeld für die E-Mail Adresse beschriften	1	1	
4	10	für das Texteingabefeld für die E-Mail Adresse eine Breite von 25 Zeichen einstellen	1	1	
4	11	für das Texteingabefeld für die E-Mail Adresse eine maximale Anzahl Zeichen (25 Zeichen) einstellen	1	2	- negativer Aspekt: Tab-Navigation innerhalb des Property Inspektors funktioniert nicht
4	12	Texteingabefeld für die E-Mail Adresse als Pflichtfeld setzen	1	1	
4	13	für das Texteingabefeld für die E-Mail Adresse die Validierung auf eine gültige Eingabe einstellen	1	2	- negativer Aspekt: Bezeichnung „Validierung" nicht unbedingt sofort klar
5	14	Optionsfeld für den Problemgrad hinzufügen	1	1	
5	15	Optionsfeld für den Problemgrad beschriften	1	1	
5	16	dem Optionsfeld für den Problemgrad die Optionen „hoch" und „niedrig" hinzufügen	1	1	
5	17	bei dem Optionsfeld für den Problemgrad die Option „niedrig" als Vorauswahl einstellen	1	1	
6	18	Auswahlliste für die Details hinzufügen	1	4	- unklar welches Formularelement zu wählen ist. nimmt Checkliste --> Hinweis notwendig
6	19	Auswahlliste für die Details beschriften	1	1	
6	20	der Auswahlliste für die Details die Werte „Hardware" und „Software" hinzufügen	1	1	
6	21	bei der Auswahlliste für die Details die Höhe 1 einstellen	1	1	
6	22	Einstellen, dass die Auswahlliste für die Details nur dann zu sehen ist, wenn der Problemgrad „hoch" ausgewählt ist.	3	2	- sucht eine Weile, wo die Bedingung einzustellen ist, findet aber von alleine die Anzeigeoptionen - musste sich bei der Regeldefinition kurz orientieren
7	23	Textbereich zur Eingabe der Problembeschreibung hinzufügen	1	1	
7	24	Textbereich zur Eingabe der Problembeschreibung beschriften	1	1	
7	25	bei dem Texteingabefeld zur Eingabe der Problembeschreibung die Höhe und Breite festlegen	1	1	
7	26	bei dem Texteingabefeld zur Eingabe der Problembeschreibung die Vorbelegung festlegen	1	4	- gibt Vorbelegung in Property ‚interne Beschreibung' ein, vermisst daraufhin Feedback. Hinweis notwendig, wo Vorbelegung einzutragen ist
8	27	E-Mail Bestätigung an den Absender des Formulars einstellen	2	2	- schaut in Formularpalette, erwartet dort aber auch keine E-Mail Bestätigung. Geht daher über obere Navigation einen Schritt weiter
8	28	Betreff der E-Mail Bestätigung an den Absender des Formulars festlegen	1	1	
8	29	Festlegen, dass der Empfänger der E-Mail Bestätigung mit seinem im Formular eingegebenen Nachnamen begrüßt wird.	1	1	
8	30	Text der E-Mail Bestätigung an den Absender des Formulars festlegen	1	1	
9	31	E-Mail zur Verarbeitung der Formulardaten einstellen	1	1	
9	32	Betreff der E-Mail zur Verarbeitung der Formulardaten festlegen	1	1	
9	33	Versandkriterium für die E-Mail zur Verarbeitung einstellen	2	1	- muss sich kurz innerhalb der Benachrichtigungskriterien orientieren

perForm: tabellarische Auswertung der Testdaten Seite 2

Task Nr.	C.D. Orientierung	C.D. Umsetzung	C.D. Bemerkungen	B.K. Orientierung	B.K. Umsetzung
1	1	1		1	1
2	3	3	- sucht wo neues Formular zu erstellen - nicht klar, dass default Layout bereits ausgewählt	1	2
3	1	1		1	1
3	1	1		1	1
3	1	1		1	1
3	1	1		1	1
3	1	1		1	1
4	1	1		1	1
4	1	1		1	1
4	1	1		1	1
4	1	1		1	1
4	1	1		1	1
4	1	1		1	2
5	1	1		1	2
5	1	1		1	1
5	1	2	- kurze Unsicherheit, ob Optionen wirklich in der entsprechenden Property untereinander eingetragen werden können	1	1
5	1	1		1	1
6	1	1		1	1
6	1	1		1	1
6	1	1		1	1
6	1	1		1	1
6	3	1	- sucht in verschiedenen Properties, findet dann aber Anzeigeoptionen	1	1
7	1	1		1	1
7	1	1		1	1
7	1	1		1	1
7	1	1		1	2
8	2	1	- orientiert sich kurz, geht dann aber in den Schritt „Bestätigung“	1	2
8	1	1		1	1
8	4	1	- sucht in den Versandkriterien, probiert Regeldefinitionen durch - Hinweis notwendig: Nachname soll im Inhalt der E-Mail stehen	1	1
8	1	1		1	1
9	4	1	- will neue E-Mail Benachrichtigung erstellen - Prinzip Bestätigung/Benachrichtigung unklar - Hinweis notwendig: nächster Schritt	1	1
9	1	1		1	1
9	1	1		3	2

perForm: tabellarische Auswertung der Testdaten Seite 3

Task Nr.	B.K. Bemerkungen	A.V. Orientierung	A.V. Umsetzung	A.V. Bemerkungen	Mittelwert Problemgrad
1		1	2	- Bedeutung des Schrittes „Verarbeitung“ nicht sofort klar	1,1
2	- findet es nicht ganz klar, dass man auf ‚Standard-Formular‘ klicken muss - findet es unklar, was er bei Schritt eins ausgewählt hat, da im zweiten Schritt kein Formular zu sehen ist.	2	1	- Link zum Erstellen des neuen Formulars im default Layout schwer zu erkennen	1,8
3		1	1		1,1
3		1	2	- Unklar ob das System einen Doppelpunkt nach Beschriftung setzt	1,4
3		1	1		1,1
3		1	1		1,0
3		1	1		1,0
4		1	1		1,0
4		1	1		1,0
4		1	1		1,0
4		1	1		1,1
4		1	1		1,0
4	- negativer Aspekt: Nicht sofort klar, was Property „Validierung“ ist	1	1		1,3
5	- überlegt kurz welches Formularelement auszuwählen ist	1	1		1,1
5		1	1		1,0
5		2	1	- nicht sofort klar, dass unter der Property „Optionen“ die Werte einzutragen sind	1,3
5		1	1		1,0
6		2	1	- überlegt kurz, welches das richtige Formularelement ist	1,5
6		1	1		1,0
6		1	1		1,0
6		1	1		1,0
6		1	1		1,6
7		1	1		1,0
7		1	1		1,0
7		1	1		1,0
7	- negativer Aspekt: keine Tab Navigation innerhalb des Properties-Inspektor möglich	1	1		1,5
8	- negativer Aspekt: in der Navigationsleiste der einezelnen Schritte ist nur der Text klickbar, schöner wäre der ganze Bereich	2	1	- muss sich erst kurz in Modellierungsprozess orientieren	1,6
8		1	1		1,0
8		2	1	- orientiert sich kurz, kommt dann aber sehr gut klar	1,5
8		1	1		1,0
9		1	1		1,4
9		1	1		1,0
9	- schaut in Verarbeitungskriterien, verlässt diese aber wieder und geht zurück zu ‚Benachrichtigung verfassen‘ - geht wieder in Benachrichtigungskriterien - negativer Aspekt: wird Groß- Kleinschreibung unterschiedne? --> Hinweis wäre schön	1	2	- leichte Unklarheit darüber, wie er es formulieren würde, wenn ja nach Anfangsbuchstaben A-M und N-Z unterschiedliche Empfänger benachrichtigt werden sollen	1,6

Grafische Darstellung der Auswertungtabelle für das Produkt *jotform*

Abbildung E.1 zeigt den Verlauf der aufgetretenen Usability-Probleme für den Aspekt der Orientierung bei den einzelnen Mikrotasks für jeden Probanden.

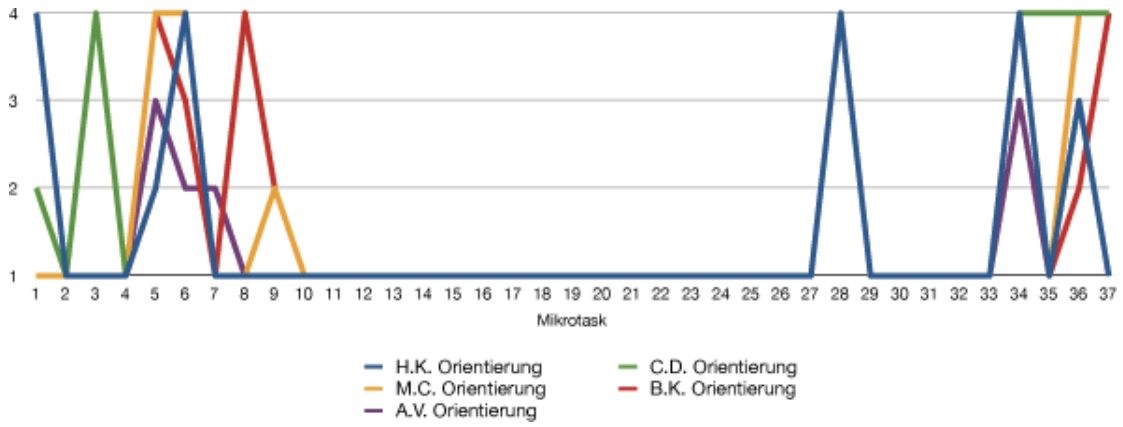


Abbildung E.1: *jotform*: Usability-Probleme der einzelnen Probanden für den Aspekt der Orientierung bei den einzelnen Mikrotasks

Abbildung E.2 zeigt den Verlauf der aufgetretenen Usability-Probleme für den Aspekt der Umsetzung der einzelnen Mikrotasks für jeden Probanden.

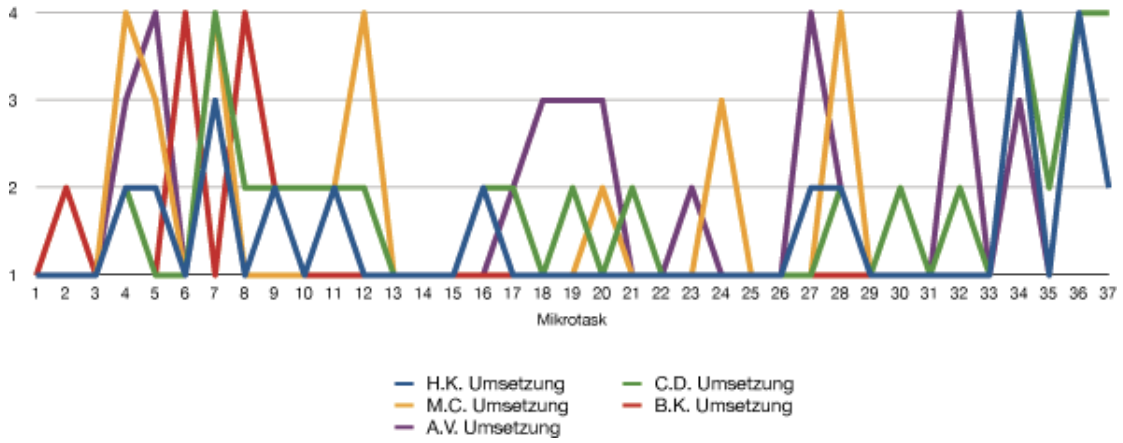


Abbildung E.2: *jotform*: Usability-Probleme der einzelnen Probanden für den Aspekt der Umsetzung bei den einzelnen Mikrotasks

Grafische Darstellung der Auswertungtabelle für das Produkt *formdesk*

Abbildung E.3 zeigt den Verlauf der aufgetretenen Usability-Probleme der einzelnen Probanden für den Aspekt der Orientierung.

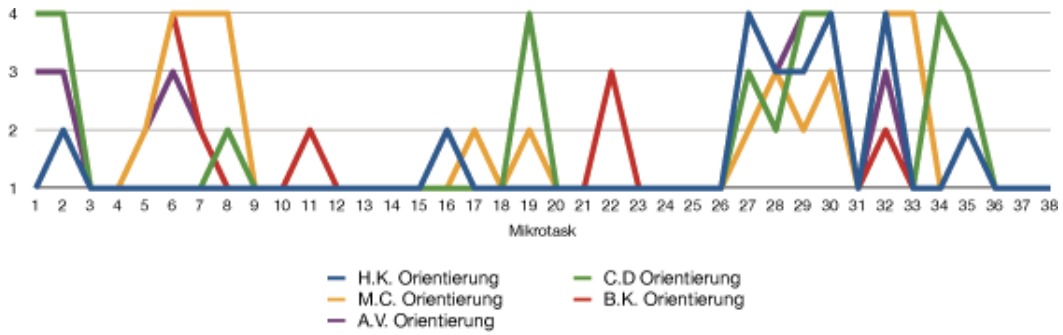


Abbildung E.3: *formdesk*: Usability-Probleme der einzelnen Probanden für den Aspekt der Orientierung bei den einzelnen Mikrotasks

Abbildung E.4 zeigt den Verlauf der aufgetretenen Usability-Probleme für den Aspekt der Umsetzung der einzelnen Mikrotasks für jeden Probanden.

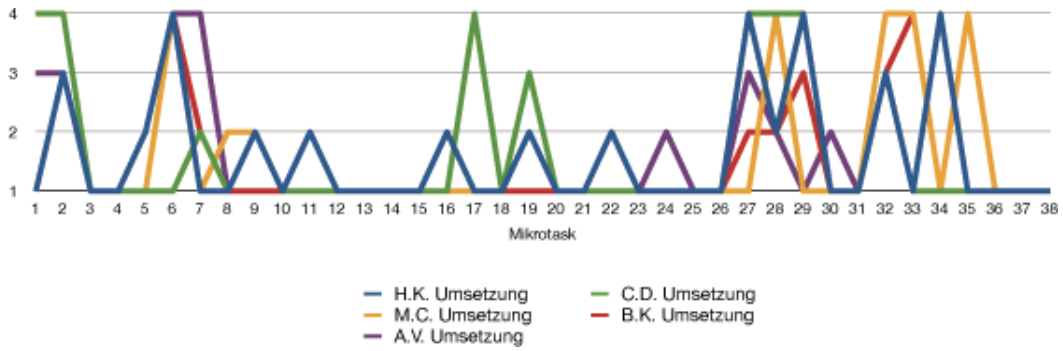


Abbildung E.4: *formdesk*: Usability-Probleme der einzelnen Probanden für den Aspekt der Umsetzung bei den einzelnen Mikrotasks

Grafische Darstellung der Auswertungtabelle für das Produkt *perForm*

Abbildung E.5 zeigt den Verlauf der aufgetretenen Usability-Probleme der einzelnen Probanden für den Aspekt der Orientierung.

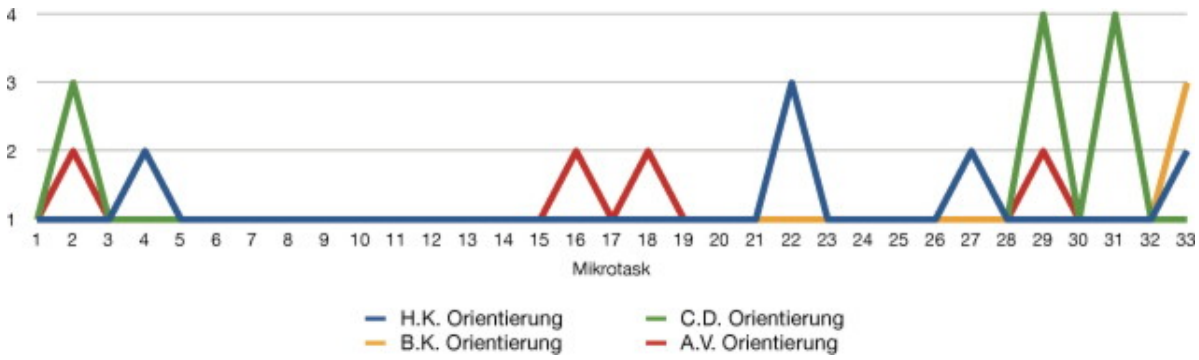


Abbildung E.5: *perForm*: Usability-Probleme der einzelnen Probanden für den Aspekt der Orientierung bei den einzelnen Mikrotasks

Abbildung E.6 zeigt den Verlauf der aufgetretenen Usability-Probleme für den Aspekt der Umsetzung der einzelnen Mikrotasks für jeden Probanden.

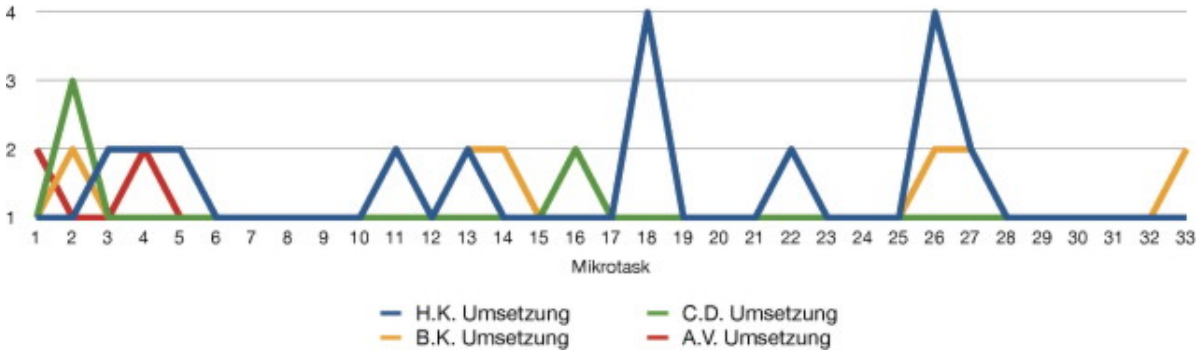


Abbildung E.6: *perForm*: Usability-Probleme der einzelnen Probanden für den Aspekt der Umsetzung bei den einzelnen Mikrotasks