Modellierung und Generierung von Benutzeroberflächen für interaktive Softwaresysteme unter der Nutzung von Mustern

Entwurf einer modell- und musterbasierten Entwicklungsumgebung:

Pattern-based Modeling and Generation of Interactive Software Systems (PaMGIS)

Dissertation

zur Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

der Fakultät für Informatik und Elektrotechnik
der Universität Rostock

vorgelegt von

Jürgen Engel, geboren am 17.12.1964 in Augsburg
aus Klosterlechfeld

Rostock, 17. Januar 2019

https://doi.org/10.18453/rosdok_id00002728

Gutachter:

Prof. Dr.-Ing. habil. Peter Forbrig
Leiter Lehrstuhl für Softwaretechnik am Institut für Informatik der Universität Rostock

Prof. Dr.-Ing. Christian Märtin
Fakultät für Informatik der Hochschule Augsburg

Univ.-Prof. Dr. rer. nat. Michael Herczeg Dipl.-Inform.
Institut für Multimediale und Interaktive Systeme der Universität zu Lübeck

Tag der Verteidigung: 5. Juni 2020

Vorwort

Die vorliegende Dissertation entstand im Rahmen einer Kooperation des Lehrstuhls für Softwaretechnik am Institut für Informatik der Universität Rostock und der Fakultät für Informatik der Hochschule Augsburg.

Die Betreuung dieser Arbeit erfolgte durch Herrn Prof. Dr.-Ing. habil. Peter Forbrig und Herrn Prof. Dr.-Ing. Christian Märtin. Für ihre stete Gesprächsbereitschaft, ihre Anregungen und konstruktive Kritik gilt ihnen mein besonderer Dank.

Für die Übernahme des dritten Gutachtens und den damit verbundenen Zeit- und Arbeitsaufwand bedanke ich mich herzlich bei Herrn Univ.-Prof. Dr. rer. Nat. Michael Herczeg.

Für sein Review und Feedback danke ich Herrn Dipl.-Inf. (FH), M.Sc. Christian Herdin, dessen Promotion auf dieser Arbeit aufbaut.

Zudem bedanke ich mich bei den beiden Master-Absolventen der Hochschule Augsburg, Herrn B.Sc. Antonios Fesenmeier und Herrn B.Sc. Roland Reitböck. Mit ihren Implementierungen wichtiger Teile des PaMGIS-Frameworks haben sie zu dessen Praxisnähe beigetragen.

Nicht zuletzt danke ich meiner Frau Monika für ihre Unterstützung als Lektorin und ihr Verständnis und die Ermutigung während der gesamten Bearbeitungszeit.

Besonderer Dank gilt auch meinem kleinen Sohn Lorenz, der häufig so tapfer auf meine Verfügbarkeit verzichten musste.

Jürgen Engel
Klosterlechfeld, im Juni 2020

Diese Arbeit ist meinen Eltern gewidmet, die mir so Vieles ermöglicht haben.

Vollkommenheit entsteht offensichtlich nicht dann,
wenn man nichts mehr hinzuzufügen hat, sondern
wenn man nichts mehr wegnehmen kann.

Antoine de Saint-Exupéry (29.06.1900 - 31.07.1944)

Französischer Schriftsteller

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation und Zielsetzung

Mit steigender Bedeutung und Verbreitung von Computern und der zunehmenden Komplexität der darauf ablaufenden Programme, die vermehrt auch Eingaben durch ihre Benutzer erforderten, stellte sich automatisch die Frage nach einer adäquaten Art und Weise, in der Mensch und Computer miteinander kommunizieren können. Einer der ersten wegweisenden Ansätze wurde von Ivan Sutherland im Jahr 1963 am *Lincoln Laboratory*[1] im Rahmen der Entwicklung des Programms *Sketchpad* entwickelt. Hiermit konnten grafische Objekte mithilfe eines sogenannten Light-Pen direkt manipuliert, also beispielsweise markiert, auf dem Bildschirm verschoben und verändert werden. Die Verwendung von grafischen Kontrollelementen für Benutzeroberflächen, im Englischen auch als *Widgets* bezeichnet, brachte William Newman 1966 am *Imperial College* in London bei seiner Arbeit an einem Programm namens *Reaction Handler* ins Spiel, indem er eine Art von grafischen Potentiometer zur Lichtregelung einführte. Weitere, bis heute zum Einsatz kommende Techniken wie Fenster (engl. Windows) und Hypertext-Links, fanden ab 1968 im *oN-Line System* (NLS) von Douglas Engelbart praktische Anwendung. Entwickelt wurde dieses System am Standford Research Institute[2]. Der Begriff der *Icons* wurde von David Canfield Smith im Jahr 1975 im Kontext seiner Doktorarbeit *Pygmalion* an der Standford University geprägt. Er machte sie später als Chef-Designer der *Xerox Star*, einer von *Xerox SDD*[3] entwickelten Workstation mit grafischer Benutzeroberfläche, populär. Ebenfalls aus den 1970er Jahren stammt die Idee des *WYSIWYG* (what you see is what you get), das im deutschsprachigen Raum auch als Echtbilddarstellung bezeichnet wird. Hierbei wird ein Dokument auf dem Bildschirm genau so angezeigt, wie es auf einem anderen Ausgabegerät, also etwa einem Drucker, dargestellt wird. Entwickelt wurde dieses Verfahren am *Xerox Palo Alto Research Center*[4] und fand beispielsweise Verwendung im Texteditor *Bravo* und Zeichenprogramm *Draw*. Als weiteres, bahnbrechendes Ereignis sei an dieser Stelle natürlich auch noch die Realisierung des *World Wide Web* (WWW) auf Basis der Hypertext-Konzepte durch die *Europäische Organisation für Kernforschung*[5] im Jahr 1990 genannt. Der erste Browser für das WWW namens *Mosaik* wurde 1993 am *National Center for Supercomputing Applications*[6] entwickelt [1].

Waren anfangs die Anforderungen eine Mensch-Computer-Interaktion (HCI[7]) und die technischen Möglichkeiten zu deren Realisierung rückblickend eher bescheiden und von untergeordneter Bedeutung, so besitzt dieses Thema heute eine sehr hohe Komplexität und

---

1 Forschungs- und Entwicklungszentrum des Verteidigungsministeriums der USA in Lexington, Massachusetts
2 Stanford Research Institute (SRI) in Menlo Park, Kalifornien, USA
3 Xerox Systems Development Division in El Segundo, Kalifornien, USA
4 Xerox Palo Alto Research Center in Palo Alto, Kalifornien, USA
5 Conseil Européen pour la Recherche Nucléaire (CERN) bei Meyrin im Kanton Genf (Schweiz)
6 National Center for Supercomputing Applications (NCSA) in Urbana, Illinois, USA
7 Human-Computer Interaction

spielt zunehmend eine entscheidende Rolle bei der Frage nach Erfolg oder Misserfolg von technischen Produkten.

Mittlerweile sind interaktive Systeme und damit auch interaktive Software kaum mehr aus dem heutigen Leben wegzudenken. Unabhängig von Zeit und Ort nutzt die Menschheit direkt oder indirekt Produkte und Dienste, die auf interaktiver Software aufbauen, sei es in Form von Anwendungen im World-Wide-Web, von Navigationssystemen, intelligenten Haushaltsgeräten oder vernetzten Fahrzeugen, Telekommunikationsplattformen, am Körper tragbare Apparaturen (sog. Wearables) oder viele andere Arten von elektronischen Geräten. Die betreffenden Endgeräte stellen hierbei schier unbegrenzte Möglichkeiten zur Interaktion, die nahezu die gesamte Sinneswahrnehmung des Menschen einbezieht, zur Verfügung. Dazu gehören beispielsweise textbasierte und graphische Kommunikation sowie Gestensteuerung im visuellen Bereich, Sprachsteuerung mittels Spracherkennung und –synthese im akustischen Bereich und Vibrationen und Blindenschrift im taktilen Bereich.

Heute sollen und müssen erfolgreiche Software-Produkte häufig auf einer Vielzahl von unterschiedlichsten Endgeräten unter Berücksichtigung derer jeweils inhärenten Möglichkeiten und Einschränkungen verfügbar sein. Dies bedeutet gegebenenfalls auch die Unterstützung mehrerer verschiedenartiger Betriebssystem- und Programmierungsumgebungen. Zudem erwarten die Käufer und Benutzer solcher Systeme ein hohes Maß an Zuverlässigkeit, Gebrauchstauglichkeit (Usability) und ein durchgängiges und ansprechendes Nutzungserlebnis (User Experience). Ferner sehen sich die System-Hersteller oft einem harten Wettbewerb ausgesetzt, der sie zu Kosteneffizienz und kurzen Produktentwicklungs- und -einführungszeiten zwingt.

Es erscheint nahezu unmöglich, allen genannten Anforderungen gleichermaßen gerecht zu werden, wenn traditionelle Software-Engineering-Techniken zum Einsatz kommen. Ein möglicher und vielversprechender Ausweg aus diesem Dilemma ist zum einen eine weitestgehend entkoppelte Erstellung der eigentlichen Anwendungslogik und der Benutzerschnittstelle und zum anderen die Nutzung eines modellgetriebenen Entwicklungsansatzes. Hierbei werden die ausschlaggebenden Aspekte des User-Interface (UI) mithilfe von Modellen auf unterschiedlichen Abstraktionsebenen spezifiziert. Diese Modelle können zu einer - zumindest teilweise - automatisierten Generierung von Benutzeroberflächen verwendet werden. Um die Erstellung und Pflege der Modelle zu erleichtern wird der modellgetriebene Ansatz um einen musterbasierten Anteil erweitert. Hierbei soll Entwurf- und Implementierungs-Know-how bezüglich der Benutzerschnittstelle so formalisiert werden, dass die (teilweise) automatisierte Generierung von Benutzerschnittstellen optimal unterstützt wird.

Diese Dissertation umfasst den Entwurf einer Entwicklungsumgebung für Benutzeroberflächen, bei der die im Software-Engineering allgemein anerkannten Techniken der modellgetriebenen und der musterbasierten UI-Entwicklung zu einem integrierten Entwicklungsansatz kombiniert werden. Ziel ist es, von den Vorteilen beider Ansätze zu profitieren und größtmögliche Wiederverwendung von Wissen und bereits erstellten Artefakten zu erreichen.

1.2. Inhaltliche Einordnung der Arbeit

In der vorliegenden Dissertation erfolgt die Herleitung und Beschreibung des Entwurfs der integrierten modellgetriebenen und musterbasierten Entwicklungsumgebung für interaktive Benutzeroberflächen *Pattern-based Modeling and Generation of Interactive Software Systems* (PaMGIS). Hierbei wird auf bereits existierende Konzepte und Forschungsergebnisse aus den genannten Bereichen zurückgegriffen. Die generelle thematische Einordnung der Arbeit wird in Abbildung 1 in Anlehnung an das *Information Systems Research Framework* nach Hevner et al. [2] dargestellt. Hierbei werden die wesentlichen Begriffe hinsichtlich der Anwendungs- bzw. Forschungsdomäne, der bereits vorhandenen und genutzten Wissensbasis und der durch die Arbeit erlangten Arbeitsergebnisse in Beziehung zueinander gesetzt.

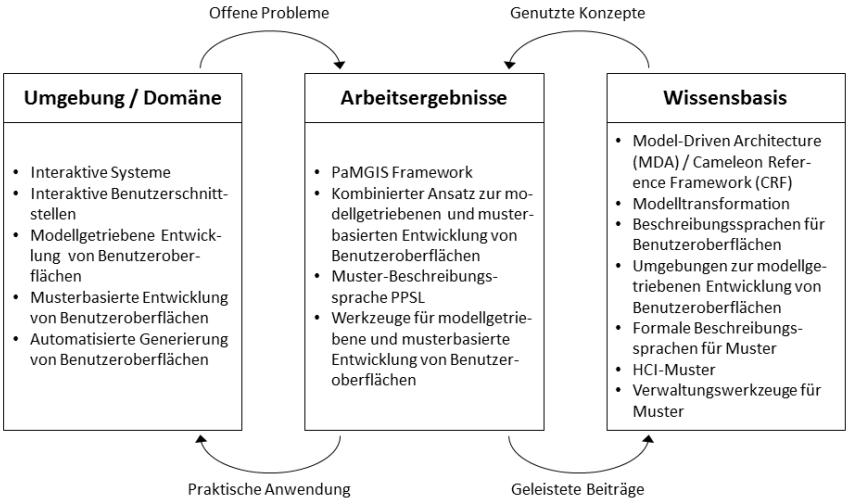


Abbildung 1 THEMATISCHE EINORDUNG DER DISSERTATION

Details zu allen bei der Recherche genutzten Quellen aus der Wissensbasis und deren Verwendung beim Entwurf des PaMGIS-Frameworks können den weiteren Kapiteln entnommen werden.

1.3. Forschungsfragen

Ausgehend von der in Kapitel 1.1 beschriebenen Zielsetzung der Dissertation in Form eines kombinierten modellgetriebenen und musterbasierten Entwicklungsansatzes für interaktive Benutzeroberflächen liegt die folgende, erste Forschungsfrage nahe:

1. *Kann eine Wiederverwendung von Design-Know-how durch den Einsatz von HCI-Patterns bei der modellgetriebenen Entwicklung von interaktiven Benutzeroberflächen realisiert werden?*

Zur Erreichung des genannten Ziels wurde zunächst der Gedanke verfolgt, bestehende modellgetriebene Entwicklungsumgebungen für Benutzerschnittstellen zu untersuchen und eine geeignete auszuwählen, die dann um einen musterbasierten Anteil erweitert werden

sollte. Dieser Plan musste aber letztlich aufgegeben werden, da zu keinem der betrachteten Frameworks vollständige und ausreichend detaillierte Informationen vorlagen. Aus diesem Grund wurde der Umfang der Dissertation entsprechend erweitert, so dass auf Basis der durch die Analysen gewonnen Kenntnisse auch der modellgetriebene Teil der kombinierten UI-Entwicklungsumgebung neu entworfen wurde. Somit wurden folgende Anschlussfragen hinsichtlich des modellgetriebenen Framework-Anteils relevant:

2. *Wie kann eine modellgetriebene Entwicklungsumgebung für Benutzeroberflächen aufgebaut sein, damit sowohl eine - zumindest teilweise - automatisierte UI-Generierung erfolgen kann als auch ein sinnvoller Einsatz von HCI-Patterns ermöglicht wird?*
3. *Welche Modelle sollen hierbei zum Einsatz kommen und in welchen Beziehungen stehen diese zueinander?*
4. *In welcher Weise können diese Modelle formal beschrieben werden, damit eine sinnvolle Kombination mit HCI-Patterns möglich ist?*

Weiterhin sind folgende zu untersuchende Fragestellungen bezüglich des musterbasierten Anteils interessant:

5. *Wie können Muster derart formalisiert beschrieben werden, dass sie einerseits maschinell zu verarbeiten sind und andererseits eine modellgetriebene Generierung von Benutzeroberflächen sinnvoll ergänzen?*
6. *Ist es möglich, bereits existierende Muster in diesen neuen Formalismus zu überführen und so zu erweitern, dass sie bei einem kombinierten Entwicklungsansatz wiederverwendet werden können?*

Hinsichtlich des angestrebten kombinierten modellgetriebenen und musterbasierten Entwicklungsansatzes sind anschließend noch folgende Fragen von zentraler Bedeutung:

7. *Aus welchen Schritten besteht die neue, kombinierte Entwicklungsmethodik und in welcher zeitlichen Abfolge stehen diese?*
8. *Welche Hilfsmittel und Werkzeuge zur Unterstützung der Benutzer des Frameworks sind hinsichtlich der kombinierten Entwicklungsmethodik sinnvoll beziehungsweise notwendig?*
9. *Führt die kombinierte Nutzung von Mustern und Modellen in einer gemeinsamen Entwurfs- und Entwicklungsumgebung zu besseren Benutzerschnittstellen oder höherer Effizienz bei deren Entwicklung im Vergleich zu reinen modellgetriebenen bzw. musterbasierten Ansätzen?*

1.4. Inhaltliche Gliederung

Im weiteren Verlauf ist die vorliegende Dissertation in vier Hauptteile mit Informationen zu folgenden Themenkomplexen gegliedert:

1. Modellgetriebene Entwicklung
2. Musterbasierte Entwicklung
3. Entwurf der Entwicklungsumgebung (PaMGIS)
4. Fallbeispiel

Detaillierte Informationen zu modellgetriebenen Entwicklungsansätzen befinden sich in Kapitel 2. Im ersten Unterkapitel werden zunächst grundlegende Definitionen und einschlägige Verfahren erklärt. Die nachfolgenden Abschnitte beschäftigen sich einerseits mit häufig zum Einsatz kommenden Modelltypen und andererseits mit den Möglichkeiten der Transformation von Modellen. Zwei weitere Unterkapitel widmen sich der Analyse und Bewertung existierender Modellierungssprachen und bestehender modellbasierter Entwicklungsumgebungen für Benutzeroberflächen (MB-UIDE[8]).

Die musterbasierte Entwicklung wird in Kapitel 3 behandelt. Auch hier werden in einem ersten Unterkapitel Begrifflichkeiten definiert und Grundlagen erläutert. Die folgenden drei Abschnitte beschäftigen sich mit der Analyse und Bewertung von formalen Beschreibungssprachen für Muster, bestehenden Pattern-Sammlungen und existierenden Pattern-Werkzeugen.

Die Ergebnisse der vorangehenden Untersuchungen fließen in den Entwurf der integrierten modellgetriebenen und musterbasierten Entwicklungsumgebung für interaktive Benutzeroberflächen *Pattern-based Modeling and Generation of Interactive Software Systems* (PaMGIS) ein, der in Kapitel 4 detailliert beschrieben wird. Das erste Unterkapitel beschäftigt sich hierbei mit grundsätzlichen Design-Entscheidungen. In den folgenden beiden Abschnitten wird zunächst die generelle Struktur des Frameworks und das zugrundeliegende Benutzerrollenmodell erklärt, während in Kapitel 4.4 auf die neue, kombinierte Entwicklungsmethodik eingeht. Detailaspekte des modellgetriebenen Teils von PaMGIS werden schließlich in Kapitel 4.6 und des musterbasierten Anteils in Kapitel 4.7 dargelegt.

In Kapitel 5 befindet sich eine Fallstudie, welche die praktische Bedeutung und Umsetzung der vorhergehenden theoretischen Ausführungen verdeutlicht. Das Fallbeispiel beschäftigt sich mit Benutzeroberflächen für Verkaufssysteme für Tickets zur Personenbeförderung; im konkreten Fall für den Bahnverkehr.

Im Kapitel 6 enthaltenen Epilog werden zunächst die Inhalte dieser Arbeit zusammengefasst. Danach erfolgt die Stellungnahme zu den eingangs beschrieben Forschungsfragen. Abschließend werden noch die Möglichkeiten hinsichtlich weitergehender Forschungsrichtungen und der Weiterentwicklung des PaMGIS-Frameworks aufgezeigt.

[8] Model-based User Interface Development Environment

2. Modellgetriebene Entwicklung

Die vorliegende Dissertation befasst sich mit dem Entwurf eines Frameworks zur Modellierung und Generierung von Benutzeroberflächen für interaktive Systeme. Die Grundidee ist hierbei, existierende Techniken und Verfahren der modellgetriebenen mit Ansätzen der musterbasierten Softwareentwicklung zu kombinieren. Das folgende Kapitel dient als Einstieg in die modellgetriebene Software- und User Interface-Entwicklung. Es erklärt wesentliche Aspekte dieser Disziplin, die für das Verständnis der Funktionsweise des später beschriebenen Frameworks erforderlich sind und die explizit oder implizit für dessen Entwurf benötigt werden.

Hierzu werden zunächst in Kapitel 2.1 die erforderlichen Grundlagen in Form von Begriffsdefinitionen und maßgeblichen Architekturen beschrieben. Das sich daran anschließende Kapitel 2.2 enthält Informationen über die bei der modellgetriebenen Entwicklung häufig eingesetzten Typen von Modellen. In Kapitel 2.3 erfolgt die Beschreibung von Modell-Transformationen, die eine wesentliche Rolle bei den modellbasierten Verfahren spielen. Danach widmet sich Kapitel 2.4 einer Anzahl von insgesamt acht existierenden Modellierungssprachen, die im Detail erklärt und gemäß einer eigens hierfür entwickelten Metrik bewertet und verglichen werden. Zum Abschluss des Themenkomplexes werden in Kapitel 2.5 insgesamt 18 bestehende modellbasierte Entwicklungsumgebungen für Benutzeroberflächen beschrieben und ebenfalls mittels einer speziell dafür definierten Metrik analysiert und bewertet.

2.1. Grundlagen

Das folgende Kapitel führt in die Grundlagen der modellgetriebenen Softwareentwicklung ein. Nach der Definition von Grundbegriffen erfolgt die Beschreibung der relevanten Referenzarchitekturen *Model Driven Architecture* (MDA), *CAMELEON Reference Framework* (CRF) und *Arch-Modell*.

2.1.1. Begriffsdefinitionen

2.1.1.1. Modell

Im Sinne der modellbasierten Softwareentwicklung ist ein Model eine Beschreibung oder Spezifikation eines Software-Systems und seiner Umgebung zu einem bestimmten Zweck [3]. Ein Modell kann auch als Abstraktion eines Systems und/oder dessen Umgebung angesehen werden [4].

Ein Modell ist eine formale Beschreibung der Funktionalität, der Struktur und des Verhaltens eines Software-Systems in einem gegebenen Kontext und aus einer bestimmten Betrachtungsweise heraus. Ein Modell wird meist durch eine Kombination aus Abbildungen und Text repräsentiert. Typischerweise wird hierbei eine formale Notation verwendet, die, falls notwendig, durch Ausdrücke in natürlicher Sprache ergänzt werden. Eine Modell-Spezifika-

tion wird als formal bezeichnet, wenn sie auf einer Sprache mit wohldefinierter semantischer Bedeutung all ihrer Sprach-Konstrukte basiert [5].

Ein Modell wird mithilfe einer Modellierungssprache beschrieben, deren Syntax und Semantik wiederum durch ein Metamodell festgelegt wird [6].

2.1.1.2. Modellbasiert, modellgetrieben

Ein Ansatz zur Software-Entwicklung wird als modellbasiert oder modellgetrieben bezeichnet, wenn Modelle als Hilfsmittel dazu genutzt werden, das Verständnis, den Entwurf, die Erstellung, die Verteilung, den Betrieb, die Wartung und die Modifikation von Software-Systemen zu leiten [3].

2.1.2. Model Driven Architecture (MDA)

Die im Jahr 1989 gegründete *Object Management Group* (OMG) ist ein offenes, internationales Konsortium für die Entwicklung von nicht-kommerziellen Technologie-Standards. Beteiligt an der Entwicklung von OMG-Standards sind Hersteller, Endanwender, akademische Einrichtungen und staatliche Behörden [7]. Die OMG wurde für das Vorhaben ins Leben gerufen, um die Einführung neuer Software-Produkte zu beschleunigen, deren Komplexität zu reduzieren und Kosten zu senken [3].

In diesem Sinne veröffentlichte die OMG im Jahr 2001 einen, *Model Driven Architecture* (MDA) genannten, standardisierten Ansatz zur Verwendung von Modellen in der Softwareentwicklung [3]. Bis zu diesem Zeitpunkt waren Modelle bereits ein weit verbreitetes und akzeptiertes Hilfsmittel in der Software-Industrie, wurden aber hauptsächlich zu Spezifikations- und Dokumentationszwecken eingesetzt. Mit der MDA wurde nun versucht, die Repräsentation von Software-Anwendungen von der Programmcode- auf eine Modell-Ebene zu heben. Um die Komplexität der verwendeten Modelle zu begrenzen, werden diese verschiedenen Abstraktionsebenen zugeordnet. Ziel der MDA ist es, ausgehend von den Modellen höchster Abstraktion Modelle niedrigerer Abstraktionsebenen schrittweise und möglichst automatisiert zu erzeugen, bis schließlich ausführbarerer Programmcode entstanden ist. Änderungen an einer vorliegenden Software erfolgen im Idealfall nicht mehr im Quellcode, sondern in einem oder mehreren der Ausgangsmodelle [8]. Eine der treibenden Kräfte hinter MDA ist die Tatsache, dass ein und dieselbe Software-Anwendung auf mehreren System-Plattformen ablaufen soll, gegebenenfalls auch auf solchen, die zum Zeitpunkt der erstmaligen Entwicklung der Anwendung noch nicht angedacht wurden oder sogar überhaupt noch nicht verfügbar waren [9].

Die MDA ist bei ihrem Erscheinen weniger ein neuer Standard, sondern vielmehr als ein moderner Softwareentwicklungsansatz zu verstehen, der auf bereits existierenden OMG-Spezifikationen aufbaut. Hierzu gehören beispielsweise die *Unified Modeling Language*

(UML)[9], *System Modeling Language* (SysML)[10], *Business Process Model and Notation* (BPMN)[11], *Meta Object Facility* (MOF)[12] und *Common Warehouse Metamodel* (CWM)[13] [5].

Eine grundlegende Eigenschaft der MDA ist ihre Fähigkeit, den vollständigen Software-Entwicklungsprozess, beginnend mit Analyse und Design über Implementierung und Test bis hin zur Komponentenzusammenführung, Software-Verteilung und Wartung, zu unterstützen [5].

Weitere Konzepte der MDA sind die sogenannten *Viewpoints*[14], *Views*[15] und Platforms[16]. Ein *Viewpoint* ist eine Abstraktionstechnik, bei der der Fokus auf bestimmte Belange eines Systems gerichtet wird und zugleich für die beabsichtigte Betrachtung irrelevante Details weggelassen werden, um eine Vereinfachung zu erzielen. Eine *View* eines Systems, oft auch *Viewpoint*-Modell genannt, ist eine Repräsentation des Systems aus der durch einen *Viewpoint* definierten Perspektive. Eine Plattform ist eine Menge von Subsystemen und Technologien, die ein zusammengehörendes Set von Funktionalitäten bereitstellt. Diese können von Anwendungen, die auf der betreffenden Plattform ablaufen, über entsprechende Schnittstellen genutzt werden, ohne sich um die Implementierungsdetails der Funktionalitäten sorgen zu müssen. Die Plattform-Unabhängigkeit eines Modells bezeichnet den Grad dessen Unabhängigkeit von bestimmten Merkmalen der betreffenden Plattform [3].

Innerhalb der MDA sind drei verschiedene *Viewpoints* spezifiziert [3]:

1. *Computation Independent Viewpoint*[17]
2. *Platform Independent Viewpoint*[18]
3. *Platform Specific Viewpoint* [19]

Der *Computation Independent Viewpoint* betrachtet den Kontext und die Anforderungen an das System, ungeachtet der Details von dessen Struktur oder Verarbeitungsmechanismen [3]. Er wird auch als konzeptueller Standpunkt bezeichnet [5].

Der *Platform Independent Viewpoint* befasst sich mit den operationellen Abläufen des Systems, ohne aber auf die Details einer bestimmten Plattform einzugehen. Es wird genau der Anteil der gesamten Spezifikation betrachtet, der sich bei der Verwendung der einen oder einer anderen Plattform nicht ändert [3]. Dieser Standpunkt wird auch als *Analysis Viewpoint* bezeichnet [5].

Der *Platform Specific Viewpoint* erweitert den *Platform Independent Viewpoint* um die Details, die für die Benutzung einer bestimmten Plattform benötigt werden [3]. Er wird auch *Design Viewpoint* genannt [5].

---

[9] Siehe http://www.omg.org/spec/UML/
[10] Siehe http://www.omg.org/spec/SysML/
[11] Siehe http://www.omg.org/spec/BPMN/index.htm
[12] Siehe http://www.omg.org/spec/MOF/
[13] Siehe http://www.omg.org/spec/CWM/
[14] Engl.: Gesichtspunkte, Standpunkte
[15] Engl.: Sichten
[16] Engl.: Plattform
[17] Engl.: EDV-unabhängiger Standpunkt
[18] Engl.: Plattformunabhängiger Standpunkt
[19] Engl.: Plattformspezifischer Standpunkt

Gemäß der drei unterschiedlichen *Viewpoints* sind drei MDA-Modelle definiert, deren Inhalte jeweils den Systembeschreibungen aus der Perspektive des zugehörigen Standpunkts entsprechen [3]:

1. *Computation Independent Model* (CIM)
2. *Platform Independent Model* (PIM)
3. *Platform Specific Model* (PSM)

Die Spezifikation der Modelle erfolgt bevorzugt unter Zuhilfenahme der UML [3].

Zudem kommen bei der MDA sogenannte *Platform Models*[20] zum Einsatz. Sie enthalten einerseits die Beschreibungen der technischen Konzepte, die die Bestandteile der Plattform und die durch sie angebotenen Dienste repräsentieren und andererseits die Spezifikationen der verschiedenen Elemente, die durch eine Anwendung genutzt werden können [3].

Neben der Verwendung von Modellen sind Transformationen, die im MDA-Jargon auch *Mappings*[21] genannt werden, ein weiteres fundamentales Konzept der *Model Driven Architecture*. Sie können manuell, semiautomatisch oder vollautomatisch erfolgen und basieren auf der Anwendung von Transformationsregeln. Diese werden grundsätzlich in *Model Type Mappings* und *Model Instance Mappings* unterschieden. Es existiert auch eine Mischform dieser beiden Transformationsarten, die als *Combined Type and Instance Mappings* bezeichnet werden [8]. *Model Type Mappings* sind hierbei Transformationsregeln, die auf der Ebene der Sprach-Konstrukte der Modellsprache definiert werden [8]. Es handelt sich um *Mappings* von einem Modell mit in der PIM-Sprache spezifizierten Typen zu einem Modell mit in der PSM-Sprache ausgedrückten Typen [3]. Im Unterschied dazu definieren die Transformationsregeln von *Model Instance Mappings* die Abbildung konkreter Instanzen innerhalb des PIM zu konkreten Instanzen im PSM [8]. Für diese Art der Transformation werden in sogenannten *Marks*[22] zusätzliche Informationen darüber festgelegt, wie genau die betreffenden Elemente des PIM zu transformieren sind. Die Markierungen sind plattformspezifisch und deshalb nicht Teil des eigentlichen PIM. Das markierte PIM bildet die Grundlage für die Erzeugung des für die betreffende Zielplattform benötigte PSM [3].

In Abbildung 2 wird eine Übersicht über die MDA-Systemmodelle der verschiedenen Abstraktionsniveaus gezeigt. Die Transformationen zwischen den Modellen sind durch Pfeile repräsentiert. Auf PSM-Ebene sind beispielhaft die Modelle für drei mögliche Plattformen dargestellt [8].

In [8] werden folgende Vorteile hinsichtlich einer Software-Entwicklung mit MDA angeführt:

- Modelle mit hohem Abstraktionsgrad ermöglichen eine kompakte Beschreibung komplexer Systeme.
- MDA ermöglicht eine rasche Anpassung eines bestehenden Software-Systems an neue Plattformen.
- Die Standardisierung ermöglicht den integrierten Einsatz von Werkzeugen unterschiedlicher Hersteller.

[20] Engl.: Plattform-Modelle
[21] Engl.: Zuordnungen
[22] Engl.: Markierungen

- Die zumindest teilweise automatisierte Transformation von Modellen und Generierung von Programmcode beschleunigen den Entwicklungsprozess.

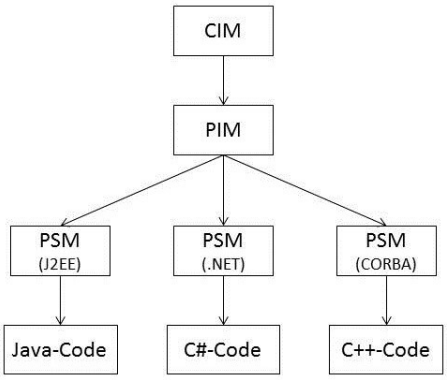


Abbildung 2 ÜBERSICHT ÜBER MDA-TRANSFORMATIONEN GEMÄß [8]

Es werden dort aber auch einige Kritikpunkte des MDA-Ansatzes aufgelistet [8]:

- Die Qualität von generiertem Programmcode hinsichtlich Strukturierung, Lesbarkeit und Wartbarkeit ist nicht zwingend sichergestellt. Falls die vollständige Generierung nicht möglich ist, können sich erforderliche, nachträglich manuell durchzuführende Änderungen am Programmcode schwierig gestalten.
- Bei manuellen Ergänzungen und Änderungen an generierten Artefakten ist die Erhaltung der Konsistenz zur Modellebene aufwändig und fehleranfällig.
- Die Performanz von generiertem Code ist nicht zwingend sichergestellt. Eine diesbezügliche Optimierung ist gegebenenfalls nur eingeschränkt möglich (siehe vorhergehenden Punkt).
- Um eine erfolgreiche Generierung zu ermöglichen, müssen sowohl die statischen als auch sämtliche dynamischen Aspekte der Anwendung modelliert werden. Dies kann zu sehr großen und komplexen Modellen führen, die schlecht zu pflegen sind.

2.1.3. CAMELEON Reference Framework

Während sich die MDA mit der modellbasierten Entwicklung vollständiger Software-Anwendungen beschäftigt und nur wenig dedizierte Unterstützung hinsichtlich der Erstellung von Benutzerschnittstellen bietet, fokussiert sich das *CAMELEON Reference Framework* (CRF) genau auf diese Disziplin.

Der Begriff CAMELEON steht hierbei für *Context Aware Modelling for Enabling and Leveraging Effective Interaction*[23] [10]. Das Framework wurde ab 2001 im Rahmen des drei Jahre laufenden CAMELEON Projekts entwickelt, das geführt durch das ISTI-C.N.R.[24] (Italien) gemeinschaftlich mit den Universitäten *Université Joseph Fourier* in Grenoble (Frankreich)

[23] Engl.: Kontext-bewusste Modellierung zur Ermöglichung und Förderung effektiver Interaktion

[24] *Istituto di Scienza e Tecnologie dell' Informazione* (Institut für Informationswissenschaft und -technologie) des *Consiglio Nazionale delle Ricerche* (nationalen Forschungsrats)

und *Université Catholique de Louvain* (Belgien) sowie den beiden Firmen *Motorola Italy* (Italien) und *IS3-GICE* (Frankreich) durchgeführt wurde [11] [12].

Das primäre Ziel des CRF ist die modellgetriebene Unterstützung der Entwicklung von sogenannten Multi-Kontext- bzw. Multi-*Target*[25]-Benutzerschnittstellen. Dies sind Benutzeroberflächen, die eine Vielzahl von Benutzungskontexten bzw. Targets unterstützen. Die unterschiedlichen Kontexte werden hierbei durch drei Dimensionen definiert [12] [13]:

1. User (Benutzer)
   sind diejenigen Personen, die das Anwendungssystem zukünftig verwenden werden oder bereits verwenden.
2. Platform (Plattform)
   sind diejenigen Hardware- und Software-Komponenten, auf denen die Anwendung später abläuft und über die der Benutzer mit dieser interagiert.
3. Environment (Umgebung)
   bezeichnet die physischen Gegebenheiten, in denen die Interaktion zwischen Benutzer und Anwendungssystem stattfinden könnte oder tatsächlich stattfindet.

Benutzerschnittstellen, die Kontextwechseln ohne Unterbrechungen standhalten und unter Beibehaltung ihrer Gebrauchstauglichkeit von ihren Benutzern auch im neuen Anwendungskontext weiterhin genutzt werden können, werden *Plastic User Interfaces*[26] genannt. Sie haben die Fähigkeit, im übertragenen Sinne „verformbar“ zu sein und sich an die neuen Gegebenheiten hinsichtlich der Eigenschaften von Benutzer, Plattform und/oder Umgebung anzupassen bzw. anpassen zu lassen. Diese Anpassbarkeit kann im Sinne des CRF sowohl bereits in der Design-Phase der Anwendungsentwicklung als auch nach Fertigstellung der Benutzerschnittstelle zur Laufzeit stattfinden [12] [13]. Zudem wird zwischen zwei Formen der Anpassungsfähigkeit unterschieden: erfolgt die Modifikation auf Initiative des Benutzers, so wird das System „adaptierbar“ genannt; erfolgt sie ohne direkten Eingriff des Users rein durch das System, so wird es als „adaptiv“ bezeichnet [12].

Das CRF definiert einen Problemraum hinsichtlich der Unterstützung von Multi-Targeting anhand dessen eine Beschreibung und Klassifizierung von Entwicklungswerkzeugen und der durch diese erzeugbaren Benutzeroberflächen vorgenommen werden kann. Er besteht aus folgenden Dimensionen [12]:

1. Target
   für die Anpassbarkeit an Benutzer, Plattformen und/oder Umgebungen
2. Prozess
   bezogen auf die unterstützten Phasen des Entwicklungsprozesses mit den Ausprägungen Design und Implementierung
3. Aktor
   bezeichnet, wer die Anpassung der Benutzeroberfläche initiieren kann - mit den Ausprägungen Mensch und System

[25] Engl.: Ziel
[26] Engl.: Plastische Benutzerschnittstellen

4. Dynamik
   der erzeugbaren Benutzeroberflächen im Sinne der Anpassungsfähigkeit – statisch vorberechnet oder dynamisch zur Laufzeit

Die Einordnung einiger existierender Entwicklungsumgebungen anhand dieser Taxonomie werden in [12] und [14] diskutiert.

Das CAMELEON Reference Framework lässt sich in drei wesentliche Teile untergliedern [12]:

1. Metamodelle
2. Design-Unterstützung
3. Laufzeit-Unterstützung

Die Metamodelle ermöglichen die Spezifikation aller wesentlichen Aspekte von Multi-Target-Benutzeroberflächen und werden in der Terminologie des CRF auch *ontologische* Modelle genannt. Sie werden zur weiteren Verwendung zu *archetypischen* und/oder *beobachteten* (*observed*) Modellen instanziiert. Archetypische Modelle sind hierbei deklarative Modelle, die die für den Design-Prozess der Benutzerschnittstellen benötigten Informationen bezogen auf einen bestimmten Kontext bereitstellen. Bei den beobachteten Modellen handelt es sich um ausführbare Modelle, die den dynamischen Anpassungsprozess zur Laufzeit unterstützen [12].

Das Design von Benutzerschnittstellen wird durch einen strukturierten Entwicklungsprozess unterstützt, anhand dessen, basierend auf den Informationen aus den archetypischen Modellen, Schritt für Schritt ausführbare User Interfaces abgeleitet werden [12].

Hinsichtlich der Laufzeit-Unterstützung werden die ausführbaren beobachteten Modelle von einer Laufzeit-Umgebung analog zu den archetypischen Modellen erzeugt. Sie bilden die Grundlage für die dynamischen Änderungen der Benutzeroberfläche Laufzeit [12].

Der grundsätzliche Aufbau des CRF mit seinen drei Hauptkomponenten wird in Abbildung 3 dargestellt. Grundsätzlich können die Informationen aller Modelle (Mitte) sowohl in allen Phasen des Entwicklungsprozesses (linke Seite) als auch für die Änderungen der Benutzerschnittstelle zur Laufzeit verwendet werden (rechte Seite). Die gestrichelten Pfeile deuten an, welche Informationen wofür bevorzugt Verwendung finden [12].

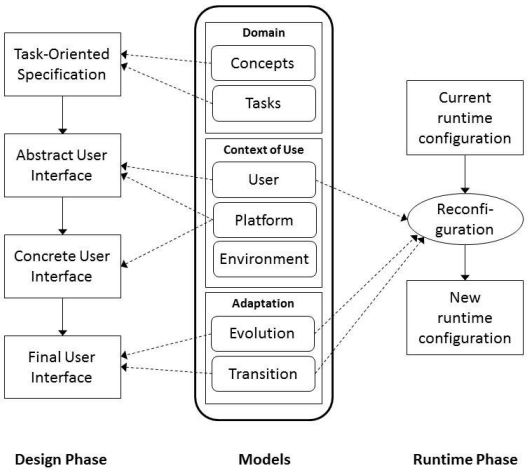


Abbildung 3 VERWENDUNG VON MODELLEN IM CAMELEON REFERENCE FRAMEWORK GEMÄß [12]

Die Modelle werden grundlegend in Domänen- (Domain), Benutzungskontext- (Context of Use) und Anpassungsmodell (Adaptation) eingeteilt. Das Domänenmodell spezifiziert die Aufgaben des Benutzers (Tasks) und diejenigen Datenelemente, die der User zur Erfüllung seiner Aufgaben lesend und/oder schreibend benötigt (Concepts). Das Benutzungskontextmodell umfasst die Teilmodelle für den Benutzer, die Plattform und die Umgebung im bereits weiter oben beschriebenen Sinn. Das Adaptionsmodell besteht schließlich aus einem Evolutionsmodell, das die neue, nach einem Kontextübergang zu verwendende Benutzerschnittstelle festlegt, und einem Transitions-Modell, das den Übergang vom bisherigen zum neuen User Interface beschreibt [12]. Weiterführende Informationen zu einigen grundlegenden Modelltypen sind in Kapitel 2.2 zu finden.

Auf der höchsten Ebene, d.h. auf dem abstraktesten Niveau bei der Erstellung einer Benutzerschnittstelle, steht die aufgabenorientierte Spezifikation (*Task-oriented Specification*), bei der die Benutzeraufgaben und die Datenelemente aus dem Domänenmodell in Relation zueinander gebracht werden. Im nächsten Schritt wird eine abstrakte Benutzerschnittstelle (*Abstract User Interface*, AUI) erzeugt [12]. Dabei handelt es sich um eine kanonische Beschreibung der Darstellung der Bestandteile des Domänenmodells in einer von konkreten Interaktoren der Zielplattform unabhängigen Art und Weise [15]. Aus diesem entsteht dann eine konkrete Benutzerschnittstelle (*Concrete User Interface*, CUI), einer Interaktor-abhängigen Beschreibung, die bereits grundsätzliche Wesenszüge der endgültigen Oberfläche aufweist (z. B. *Look&Feel*[27]), aber ausserhalb der Entwicklungsumgebung noch nicht ablauffähig ist. Letztlich wird aus dem CUI die finale Benutzerschnittstelle (Final User Interface, FUI) abgeleitet [12]. Es liegt als Quellcode der Zielprogrammiersprache vor und kann anschließend interpretiert oder kompiliert ausgeführt werden [12] [15].

Die Überführung einer Beschreibung in eine andere wird als Transformation oder auch *Mapping*[28] bezeichnet. In Abbildung 4 sind die möglichen Arten von Transformationen dargestellt [12].

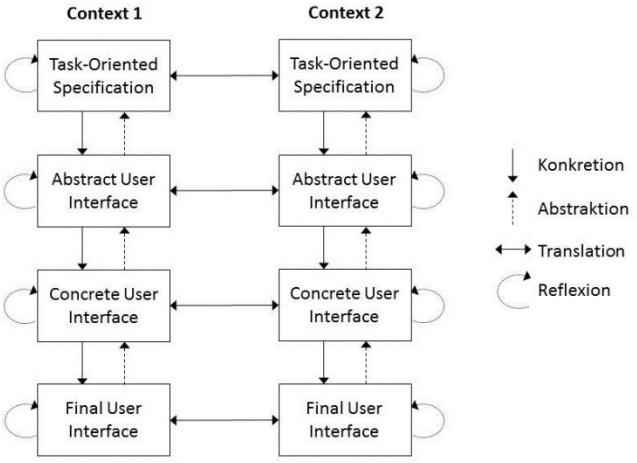


Abbildung 4 ÜBERSICHT ÜBER DIE TRANSFORMATIONEN IM CAMELEON REREFENCE FRAMEWORK GEMÄß [12]

[27] Engl.: Anmutung
[28] Engl.: Zuordnung, Abbild

Folgende Transformationen sind definiert [12] [15]:

1. Konkretion (*Reification*)
2. Abstraktion (*Abstraction*)
3. Translation
4. Reflexion (*Reflexivity*)

Eine Konkretion ist die Transformation einer Beschreibung in eine andere Beschreibung auf weniger abstraktem Niveau [15]. Sie sind in Abbildung 4 durch senkrecht nach unten gerichtete Pfeile dargestellt. FUIs können nicht weiter konkretisiert werden.

Eine Abstraktion erzeugt aus einer weniger abstrakten Beschreibung eine abstraktere [12]. Sie sind in der Abbildung als senkrecht nach oben gerichtete, gestrichelte Pfeile repräsentiert. Aufgabenorientierte Spezifikationen werden im Rahmen des CRF-Entwicklungsprozesses nicht weiter abstrahiert.

Translationen ermöglichen ein Mapping von einer UI-Beschreibung eines Kontexts zu einer anderen auf gleichem Abstraktionsniveau, die aber einem anderen Kontext zuzuordnen ist [12]. Sie sind in der Abbildung als waagerechte, bidirektionale Pfeile dargestellt.

Eine Reflexion ist eine Abbildung einer UI-Repräsentation beliebigen Abstraktionsniveaus auf eine andere gleicher Abstraktionsebene im gleichen Kontext [15]. Reflexionen sind in der Abbildung als rückführende Pfeile dargestellt.

Die Anpassung einer Benutzerschnittstelle zur Laufzeit wird im CRF in drei grundlegenden Schritten durchgeführt [13]:

1. Erkennen der Situation
2. Bestimmung einer Reaktion
3. Durchführen der Reaktion

Die detaillierte Abfolge dieses Adaptionsprozesses sind in Abbildung 5 grafisch dargestellt [13].

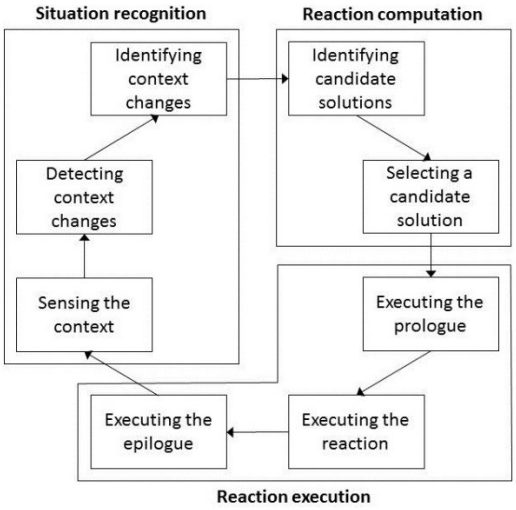


Abbildung 5 CRF-PROZESS ZUR ANPASSUNG VON BENUTZERSCHNITTSTELLEN ZUR LAUFZEIT GEMÄß [13]

Für die Erkennung der Situation ist es notwendig, den aktuellen Benutzungskontext zu „messen“ bzw. wahrzunehmen. Ferner müssen Kontextänderungen erkannt und schließlich

eindeutig identifiziert werden. Die Identifikation einer Kontextänderung kann Auslöser für eine Veränderung an der Benutzeroberfläche sein. Hierzu werden im Folgeschritt zunächst mögliche Reaktionen identifiziert und dann eine davon ausgewählt. Die Durchführung der Reaktion erfolgt dann in drei Teilschritten. Anhand eines sogenannten Prologs wird die Reaktion vorbereitet. Es wird beispielsweise die aktuell bearbeitete Aufgabe beendet, suspendiert oder abgebrochen und der zugehörige Ausführungskontext abgespeichert. Dann wird die eigentliche Änderung an der Benutzerschnittstelle durchgeführt und schließlich durch einen Epilog abgeschlossen, d.h. der zuvor gespeicherte Ausführungskontext wird wiederhergestellt [12].

Als Basis für die Realisierung dynamischer Änderungen an Benutzerschnittstellen von interaktiven Systemen zur Laufzeit wird im Rahmen des CRF die Verwendung des Referenz-Architektur-Modells mit dem Namen *ARCH-Modell* (siehe Kapitel 2.1.4) empfohlen [12].

2.1.4. Arch-Modell

Das *Arch*[29]-Modell ist ein Referenz-Architektur-Modell für Laufzeit-Umgebungen interaktiver Anwendungssysteme. Im Jahr 1991 wurde eine detaillierte Analyse von Daten, die zwischen den Benutzerschnittstellen und den Kernsystemen interaktiver Anwendungen ausgetauscht werden, veröffentlicht [16]. Auf dieser Basis erfolgte dann die Entwicklung des ARCH-Modells im *UIMS*[30] *Tool Developers' Workshop* im Rahmen der CHI[31]-Konferenz 1991 und dessen Verbesserung und Veröffentlichung im Jahr 1992. Das hauptsächlich verfolgte Ziel war die Minimierung des Anpassungsbedarfs der interaktiven Systeme bei sich ändernden Technologien auf Seiten der Benutzeroberflächen [17]. Das Arch-Modell wird im CAMELEON Reference Framework (siehe Kapitel 2.1.3) als Architektur für die Laufzeitumgebung empfohlen [12].

Das Arch-Modell sieht folgende fünf Komponenten vor [17]:

1. Domänenspezifische Komponente (Domain-specific Component)
2. Interaktionswerkzeug-Komponente (Interaction Toolkit Component)
3. Dialog-Komponente (Dialogue Component)
4. Präsentations-Komponente (Präsentation Component)
5. Domänen-Adapter-Komponente (Domain Adaptor Component)

In Abbildung 6 sind die Beziehungen zwischen den genannten Komponenten dargestellt [17]. Hierbei bilden die beiden erstgenannten Komponenten bildlich gesehen die Fundamente bzw. die Enden und die dritte die Spitze des Bogens.

[29] Engl.: Bogen, Gewölbe
[30] User Interface Management System
[31] Jährliche Konferenz zum Thema *Human-Computer Interaction* (HCI)

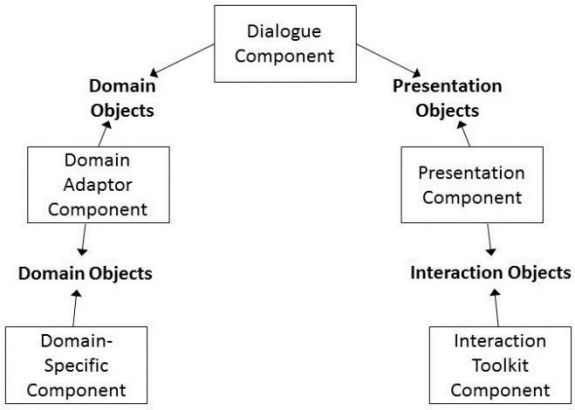


Abbildung 6 AUFBAU DES ARCH-MODELLS GEMÄß [17]

Die domänenspezifische Komponente fragt Daten der Anwendungsdomäne bzw. Geschäftslogik ab und kontrolliert und verarbeitet diese [17]. Im Kontext des CAMELEON Reference Frameworks (CRF) wird diese Komponente auch als funktionaler Kern der Anwendung (*Application Functional Core*) bezeichnet, der die Domänenkonzepte und –funktionen implementiert [12].

Die Interaktionswerkzeug-Komponente implementiert die physikalische Interaktion mit dem Endanwender mittels Soft- und Hardware [17]. Sie wird gemäß der CRF-Terminologie auch physische Präsentations-Komponente (*Physical Presentation Component*) genannt [12].

Die Dialog-Komponente regelt die Ablaufreihenfolge hinsichtlich der Benutzeraufgaben und der relevanten Domänen-immanenten Abläufe. Zudem stellt sie die Konsistenz verschiedener Sichten auf gleiche Daten und die korrekte Abbildung zwischen Domänen- und Benutzeroberflächen-spezifischen Formalismen sicher [17].

Der Datenaustausch zwischen diesen drei Kernkomponenten erfolgt nicht direkt, sondern entkoppelt über Adapter [12]. In diesem Sinne kapselt die Präsentations-Komponente die Dialog-Komponente von den Gegebenheiten der Zielplattform, in dem sie abstrakte, Toolkit-unabhängige Interaktionsobjekte zur Verfügung stellt [17]. Im CRF wird die Präsentations-Komponente auch als *Virtual Toolkit* bezeichnet [12].

Die Domänen-Adapter-Komponente trennt schließlich die Dialog- von der domänenspezifischen Komponente. Hier werden Funktionen bereitgestellt, die nicht im funktionalen Kern enthalten sind, aber für die Benutzerinteraktion benötigt werden. Zudem werden durch die Domäne vorgegebene Dialoge angestoßen, Daten reorganisiert und semantische Fehler erkannt und berichtet [17]. Im Kontext des CRF wird diese Komponente auch virtuelle Applikations-Schicht (*Virtual Application Layer*) genannt [12].

Zwischen den Komponenten werden Datenobjekte unterschiedlicher Art ausgetauscht. Der Begriff „Objekt“ bezieht sich hierbei nicht auf formale, instanziierte Objekte im Sinne der objektorientierten Programmierung, sondern ist vielmehr als abstrakte Bezeichnung für Kommunikationsmechanismen zu verstehen. Es wird zwischen Domänen-Objekten (*Domain Objects*), Präsentations-Objekten (*Presentation Objects*) und Interaktions-Objekten (*Interaction Objects*) unterschieden. Domänen-Objekte, die zwischen Domänen-Adapter- und Dialog-Komponente ausgetauscht werden, sind um Aspekte für die Verwendung in der Be-

nutzeroberfläche angereichert. Präsentations-Objekte sind virtuelle Interaktions-Objekte, die einerseits die Daten, die dem Anwender präsentiert werden, und andererseits auch die durch den Benutzer generierte Ereignisse (*Events*) umfassen ohne jedoch die konkrete Repräsentation und Art der Event-Generierung festzulegen. Die Interaktions-Objekte beziehen sich schließlich konkret auf das verwendete Toolkit der Zielplattform [17].

2.2. Grundlegende Modelle

Modellbasierte Entwicklungsansätze verwenden ein oder mehrere deklarative Modelle, mithilfe derer verschiedene Aspekte von Benutzerschnittstellen beschrieben werden. Es ergeben sich jedoch Unterschiede sowohl hinsichtlich der Terminologie, als auch der Art und Anzahl und den jeweiligen konkreten Inhalten der Modelle (siehe Kapitel 2.5). Häufig eingesetzte Modelle sind Aufgaben-, Domänen-, Dialog-, Präsentations- und Benutzer-Modelle. Diese werden in [18] ausführlich diskutiert. Die dort angeführten Definitionen sind im vorliegenden Kapitel zusammengefasst und kommentiert.

2.2.1. Aufgaben-Modell

Ein Aufgaben-Modell (engl.: *Task Model*) umfasst alle Aufgaben, die zur Erreichung eines Zieles erledigt werden müssen.

In [18] werden folgende Definitionen hierzu angeführt:

1. Eine Aufgabe ist ein Ziel zusammen mit einer geordneten Menge von Aufgaben und Aktionen, die im gegebenen Kontext zur Erfüllung des Zieles führen.
2. Ein Ziel ist die Absicht, den Status eines Artefakts zu ändern oder zu erhalten.
3. Eine Aktion ist jegliche Handlung, die eine Auswirkung auf die Änderung oder Erhaltung des Zustands eines Artefakts hat.
4. Ein Artefakt ist ein realer Gegenstand aus der gegebenen Domäne, der im Kontext der Aufgabenerfüllung Bedeutung hat.

Aufgaben sind gemäß der Definition in Unteraufgaben zerlegbar, zwischen denen kausale und/oder temporale Abhängigkeiten bestehen. Artefakte sind Objekte, die in einem Domänen-Modell modelliert werden.

Um eine Aufgabe vollständig beschreiben zu können, müssen folgende Informationen vorliegen [18]:

1. ein Ziel
2. eine nicht-leere Menge von Aktionen oder anderen Aufgaben, die zur Erreichung des Zieles benötigt werden
3. ein Konzept, wie Aktionen und/oder Aufgaben ausgewählt werden
4. das Modell eines Artefakts, das durch die Aufgabe beeinflusst wird

Es besteht eine enge Wechselbeziehung zwischen Aufgaben- und Domänen-Modell.

2.2.2. Domänen-Modell

Domänen-Modelle ähneln stark den Modellen, die aus dem Software-Engineering bekannt sind und umfassen die das User Interface betreffende Informationen. Im Wesentlichen repräsentieren sie die wichtigsten Entitäten der Anwendung, oft in Form von Klassenhierarchien für Objekte zusammen mit deren Eigenschaften. Sie umfassen zudem Beschreibungen der auf den Objekten ausführbaren Aktionen, die hierfür erforderlichen Parameter, sowie Vor- und Nachbedingungen für die Aktionen. Domänen-Modelle bilden häufig eine zentrale Rolle bei der Kommunikation zwischen den Entwicklern der eigentlichen Applikation und denen der Benutzerschnittstelle [18]. Domänen-Modelle und Aufgaben-Modelle sind typischerweise eng miteinander verbunden, da die meisten Aufgaben mit der Anzeige oder Manipulation von Daten einhergehen. Alternativ werden Domänen-Modelle auch Applikations-Modelle (engl. application models) oder Konzept-Modelle (engl. concepts models) genannt [19]. Auch im Kontext des CRF (siehe Kapitel 2.1.3) wird diese Modellart als *Concepts Model* bezeichnet. Der Begriff des Domänen-Modells wird hierbei dazu verwendet, um Aufgaben- und Konzept-Modell zusammenzufassen [12].

2.2.3. Dialog-Modell

Dialog-Modelle beschreiben die syntaktische Struktur der Mensch-Computer-Interaktion. Es beschreibt, wann der Benutzer Befehle ausführen, Auswahlen treffen und Eingaben machen kann und wenn das System Ausgabeinformationen präsentiert [18] [20] [21]. Ein Dialog-Modell repräsentiert wesentliche Informationen hinsichtlich der statischen und dynamischen Struktur der Kommunikation zwischen Benutzer und System [22]. Dialog-Modelle stehen in enger Beziehung zu Aufgaben-Modellen, da die Aufgaben verschiedenen Dialog-Zuständen im Dialog-Modell zugeordnet sind [20].

Ein Dialog-Modell unterstützt die Trennung der Semantik der eigentlichen Domänen-Anwendung und dem Dialog zwischen Mensch und Computersystem. Das Modell sollte dafür sinnvollerweise in die beiden Blöcke der Konversation (engl. conversation) und der Präsentation (engl. presentation) unterteilt werden [23] [24]. Die Konversation bezieht sich hierbei auf das Management der Interaktionen zwischen Anwendung und Benutzer auf oberem und mittlerem Abstraktionsniveau, während die Präsentation für die geeignete Anordnung und Darstellung der Elemente der Benutzeroberfläche Sorge trägt. Auf diese Weise erfolgt eine Separierung der dynamischen Aspekte des Dialogs in der Konversation und den statischen Anteilen in der Präsentation [24].

2.2.4. Präsentations-Modell

Ein Präsentations-Modell beschreibt die letztlich auf dem Bildschirm dargestellten Konstrukte mit ihren Eigenschaften und ihren Beziehungen untereinander. Man unterscheidet hierbei zwischen statischen und dynamischen Anteilen. Zum statischen Teil gehört beispielsweise die Präsentation von Standard-Widgets wie Schaltflächen oder Menüs. Abgesehen von Statusänderungen wie sichtbar/unsichtbar oder aktiv/inaktiv bleiben sie zur Laufzeit unverändert. Zum dynamischen Anteil gehört die Darstellung von anwendungsabhängigen Daten, deren Aufbau sich zur Laufzeit ändert oder erst ergibt [18].

2.2.5. Benutzer-Modell

Ein Benutzer-Modell beschreibt die für die Nutzung eines zu entwickelnden interaktiven Systems relevanten Eigenschaften der Anwender oder Gruppen von Anwendern. Letztere werden häufig auch als Rollen bezeichnet. Der hauptsächliche Zweck von User-Modellen ist die Individualisierung von Benutzerschnittstellen. Die Benutzereigenschaften können in anwendungsabhängige und anwendungsunabhängige Charakteristika eingeteilt werden. Anwendungsunabhängig sind beispielsweise Vorlieben, Fähigkeiten oder mögliche geistige oder körperliche Behinderungen. Anwendungsabhängig dagegen sind Ziele oder Fach- und Anwendungswissen. Benutzer-Modelle können auch Regeln beinhalten, mit denen aus definierten Benutzereigenschaften bestimmte Designentscheidungen, wie z. B. die Auswahl geeigneter Interaktionsobjekte, abgeleitet werden können [18].

2.3. Modell-Transformationen

Das folgende Kapitel beschäftigt sich mit Modelltransformationen, die unter anderem essenzielle Bestandteile sowohl der *Model Driven Architecture* (MDA) als auch des *CAMELEON Reference Frameworks* (CRF) sind. Zunächst werden im folgenden Unterkapitel einige grundlegende Begrifflichkeiten hierzu definiert. In Kapitel 2.3.2 erfolgt die Betrachtung der verschiedenen Typen von Transformationen, während Kapitel 2.3.3 abschließend Informationen zu bestehenden Transformationsansätzen beinhaltet.

2.3.1.1. Transformation

Die Bedeutung des Begriffs der Transformation im Kontext der Model Driven Architecture (MDA) wird von der *Object Management Group* (OMG) in [3] wie folgt festgelegt:

Eine Modell-Transformation ist der Prozess der Konvertierung eines Modells in ein anderes desselben Systems.

In [25] erfolgt eine Präsizierung dieser Aussage mittels folgender drei Definitionen:

1. Eine Transformation ist die automatische Generierung eines Zielmodells (engl. target model) aus einem Quellmodell (engl. source model) gemäß einer Transformationsdefinition (engl. transformation definition). Hierbei soll, sofern es die Beschreibungssprache des Zielmodells zuläßt, die Semantik des Quellmodells möglichst erhalten bleiben.
2. Eine Transformationsdefinition ist eine Menge von Transformationsregeln, die in ihrer Gesamtheit beschreiben, wie ein Modell in der Quellsprache (engl. source language) in ein Modell in der Zielsprache (engl. target language) transformiert werden kann.
3. Eine Transformationsregel ist eine Beschreibung, wie ein oder mehrere Konstrukte in der Quellsprache in ein oder mehrere Konstrukte in der Zielsprache transformiert werden können.

Für die Zwecke der vorliegenden Dissertation möchte ich diese Definitionen in gewisser Weise abwandeln bzw. präsizieren. Einerseits ist die in der ersten Definition geforderte, automatisierte Ausführung von Transformationen aus Gründen der Verarbeitungsgeschwindigkeit und Fehlervermeidung natürlich sinnvoll und erstrebenswert, aber nicht in allen

Fällen zwingend erforderlich. Zumindest sollen manuelle, gegebenenfalls auch interaktive Eingriffe in die Transformationen möglich sein. Andererseits wird in den beiden folgenden Definitionen auf die Quell- bzw. Zielsprachen Bezug genommen. Da Quell- und Zielmodelle einer Transformation nicht zwingend in verschiedenen Beschreibungssprachen spezifiziert sein müssen (siehe Kapitel 2.3.1.2), sind zur Verdeutlichung diese Definitionen entsprechend anzupassen. Zudem wird der Tatsache, dass Transformationen auch mehrere Quell- und/oder Zielmodelle gleichzeitig bedienen können, nicht Rechnung getragen. Somit wird der Begriff der Transformation nun wie folgt definiert:

1'. Eine Transformation ist die Erzeugung eines oder mehrerer Zielmodelle (engl. target models) aus einem oder mehreren Quellmodellen (engl. source models) gemäß einer Transformationsdefinition (engl. transformation definition). Hierbei soll, sofern es die Beschreibungssprache der Zielmodelle zuläßt, die Semantik der Quellmodelle möglichst erhalten bleiben.
2'. Eine Transformationsdefinition ist eine Menge von Transformationsregeln, die in ihrer Gesamtheit beschreiben, wie ein oder mehrere Quellmodelle in ein oder mehrere Zielmodelle überführt werden können.
3'. Eine Transformationsregel ist eine Beschreibung, wie ein oder mehrere Konstrukte des/der Quellmodell(e) in ein oder mehrere Konstrukte des/der Zielmodell(e) überführt werden können.

Transformationen werden mithilfe von Transformationssprache (engl. transformation languages) beschrieben [4] [6] [26].

2.3.1.2. Endogene und exogene Transformationen

In [6] werden Transformationen zwischen Modellen, die mithilfe derselben Modellierungssprache beschrieben sind, als endogene Transformationen bezeichnet. Exogene Transformationen sind demnach Transformationen zwischen Modellen mit unterschiedlichen Modellierungssprachen.

In [4] wird diese Unterscheidung auf Basis der Metamodelle getroffen. Demnach sind Quell- und Zielmodelle von endogenen Transformationen konform zu demselben Metamodell, während sie bei exogenen Transformationen verschiedenen Metamodellen genügen.

Alternativ werden auch die Begriffe *Umformung* (engl. rephrasing) für endogene und *Übersetzung* (engl. translation) für exogene Transformationen verwendet [4] [6].

2.3.1.3. Horizontale und vertikale Transformationen

Horizontale Transformationen (engl. horizontal transformations) sind Transformationen, bei denen sich Quell- und Zielmodell auf derselben Abstraktionsebene befinden. Transformationen zwischen Modellen unterschiedlichen Abstraktionsgrades heißen vertikale Transformationen (engl. vertical transformations) [6].

2.3.1.4. Unidirektionale und bidirektionale Transformationen

Unidirektionale Transformationen können nur in einer Richtung ausgeführt werden, d.h. das Zielmodell wird auf Basis der Informationen im Quellmodell erzeugt oder aktualisiert. Bidirektionale Transformation können hingegen in beide Richtungen angewendet werden,

was im Rahmen von Modellsynchronisationen besonders hilfreich ist. Die Bidirektionalität kann entweder durch bidirektionale Regeln oder durch Paare von komplementären, unidirektionalen Regeln für jede Richtung erreicht werden [26].

2.3.1.5. Modellmodifikation

Transformationen, bei denen Quell- und Zielmodelle identisch sind, werden Modellmodifikationen (engl. in-place transformations) genannt [4] [26].

2.3.1.6. Transformationswerkzeug

Der Begriff des Transformationswerkzeugs wird in [25] wie folgt definiert:

Ein Transformationswerkzeug dient zur automatisierten Ausführung von Transformationen auf bestimmten Quellmodellen [25].

Eine derartige Komponente, die vorgegebene Transformationen auf Quellmodellen automatisiert ausführt und dadurch die Zielmodelle erzeugt, wird in [4] als *Transformation Engine* bezeichnet.

Im Kontext der vorliegenden Dissertation wird der Funktionalität eines Transformationswerkzeugs neben der eigentlichen Ausführung der Transformationen gegebenenfalls auch die Möglichkeit zur Erstellung und Pflege von Transformationsbeschreibungen zugeordnet.

2.3.2. Arten von Transformationen

Das grundlegende Konzept von Modelltransformationen ist in Abbildung 7 visualisiert. In dieser Darstellung wird vereinfachend davon ausgegangen, dass genau ein Quellmodell in genau ein Zielmodell transformiert wird. Diese beiden Modelle sind jeweils konform zu ihren jeweiligen Metamodellen, auf die sich die Transformationsdefinition bezieht. Die Ausführung der Transformation erfolgt mithilfe der *Transformation Engine*, die das Quell- in das Zielmodell überführt [4].

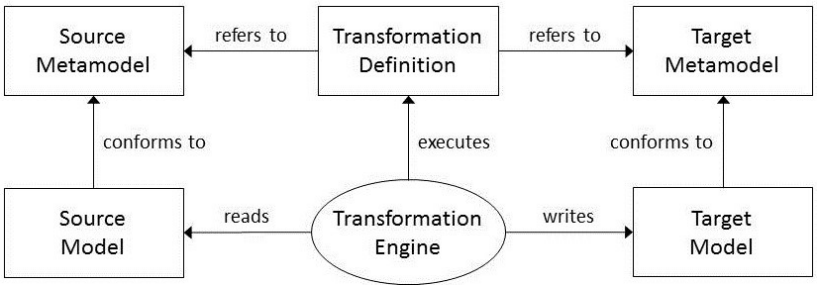


Abbildung 7 GRUNDLEGENDES KONZEPT DER MODELL-TRANSFORMATION [4]

Die in den Unterkapiteln 2.3.1.2 und 2.3.1.3 definierten Transformationseigenschaften der Vertikalität bzw. Horizontalität und der Endo- bzw. Exogenität sind linear unabhängig und können somit als zueinander orthogonale Dimensionen angesehen werden. Sie sind in Tabelle 1 einander gegenübergestellt [6].

Tabelle 1 ORTHOGONALE DIMENSIONEN VON MODELLTRANSFORMATIONEN [6]

	horizontal	**vertikal**
endogen	Neuordnung	Verfeinerung
exogen	Sprachmigration	Code-Erzeugung

Endogene, horizontale Transformationen sind als Neuordnung des Quellmodells (engl. refactoring) interpretierbar, während endogene, vertikale Transformationen dessen Verfeinerung (engl. refinement) bedeutet. Exogene, horizontale Transformationen entsprechen einer Übersetzung in eine andere Beschreibungssprache (engl. language migration) und exogene, vertikale Transformationen entsprechen einer Code-Erzeugung (engl. code generation) [6]. Bei letzterem erfolgt sowohl ein Wechsel der Sprache als auch des Abstraktionsniveaus. Für die Code-Generierung trifft zwar beides zu, aber es ist nur als Beispiel für eine solche Transformation zu verstehen, da diese auch zwischen höheren Abstraktionsebenen stattfinden kann. Zudem bleibt in Tabelle 1 unberücksichtigt, dass natürlich auch Transformationen von einer konkreteren zu einer abstrakteren Ebene, also Abstrationen, vertikale Transformationen sind.

Für die Vielzahl der existierenden Transformationsansätze gibt es eine ganze Reihe von Klassifizierungsvorschlägen, die auf grundsätzlichen Eigenschaften der jeweiligen Transformationsmechanismen beruhen, z. B. in [4] [6] [26] [27].

Generell lassen sich die Transformationsansätze in die drei Kategorien der Modell-zu-Modell-Transformationen (M2M), Modell-zu-Text-Transformationen (M2T) [4] [26] und Text-zu-Modell-Transformationen (T2M) einteilen [4]. Auf diese wird in den folgenden Unterkapiteln detaillierter eingegangen.

2.3.2.1. Modell-zu-Modell-Transformationen

Bei Modell-zu-Modell-Transformationen (engl. model-to-model, M2M) wird, wie in Abbildung 7 illustriert, das Zielmodell als Instanz des Zielmetamodells erstellt [4]. Es handelt sich prakisch um eine Übersetzung zwischen Quell- und Zielmodellen, die ihrerseits Instanzen desselben Metamodells oder verschiedener Metamodelle sind. Im Kontext der MDA werden M2M-Transformationen häufig zur Erstellung eines PSM aus einem PIM verwendet [26], im Rahmen des CRF zur Generierung eines CUI aus einem AUI.

In [4] und [26] werden bezüglich der M2M-Transformationen folgende Kategorien von Verfahren unterschieden: direkte Manipulation (engl. direct manipulation), strukturgetriebene (engl. structure-driven) Ansätze, relationale (engl. relational) Ansätze, Graphtransformation (engl. graph transformation), hybride (engl. hybrid) Ansätze und sonstige Verfahren. Zusätzlich werden in [4] noch operative (engl. operational) und schablonenbasierte (engl. template-based) Ansätze angeführt.

Ansätze zur direkten Manipulation stellen interne Repräsentation der Modelle und Programmierschnittstellen (engl. Application Programming Interface, API) zur Verfügung [4] [26]. Mithilfe einer Programmiersprache kann mithilfe der API auf die Modelle und zugehörigen Metamodelle zugegriffen werden. Es handelt sich somit um imperative Verfahren [28]. Die Implementierung der Transformationsregeln und aller anderen für die Umformung benötig-

ten Mechanismen müssen von Grund auf durch den Benutzer, d.h. durch einen Programmierer, erfolgen. Ein typischer Vertreter dieser Kategorie ist *Java Metadata Interface*[32] (JMI) [4] [26].

Bei strukturgetriebenen Ansätzen erfolgen Transformationen in zwei getrennten Phasen. Die erste Phase hat die Erzeugung der hierarchischen Struktur des Zielmodells zum Gegenstand, während in der zweiten die Attribute und Referenzen im Zielmodell gesetzt werden. Der Benutzer ist nur für die Spezifikation der Transformationsregeln verantwortlich. Die Ausführungsreihenfolge sowie die eigentliche Ausführung der Transformationen erfolgt automatisiert durch das Transformationswerkzeug. Ein Beispiel für strukturgetriebe Transformation ist das Framework *OptimalJ*[33], bei dem zunächst Elemente des Quellmodells in das Zielmodell kopiert werden und anschließend deren Modifikation erfolgt [4] [26].

Operative Transformationsverfahren haben starke Ähnlichkeit zu den Ansätzen zur direkten Manipulation, bieten aber eine verbesserte Unterstützung hinsichtlich der Definition und Durchführung von Transformationen. Hierfür wird typischerweise der Formalismus zur Metamodellierung um Ausdrucksmöglichkeiten für die Ausführung von Berechnungen erweitert. Ein Vertreter dieser Kategorie ist das *Framework Metamodeling Kernel*[34] (Kermeta) [4].

Schablonenbasierte M2M-Transformationen verwenden sogenannte Schablonen, die aus Fragmenten des Zielmodells mit eingebettetem Metacode zur Berechnung der variablen Anteile der resultierenden Template-Instanzen bestehen. Die Beschreibung der Schablonen erfolgt gewöhnlicherweise in der konkreten Syntax der Zielsprache. Der Metacode liegt meist in Form von Annotationen zu Modellelementen vor. Er besteht aus Bedingungen (engl. conditions), Iterationen (engl. iterations) und/oder Ausdrücken (engl. expressions) und wird häufig in der Metasprache *Object Constraint Language*[35] (OCL) spezifiziert [4]. Ein konkretes Beispiel für schablonenbasierte Transformationen ist in [29] beschrieben.

In die Gruppe der relationalen Transformationsverfahren fallen alle deklarativen Transformationsansätze, die auf mathematischen Relationen beruhen. Die grundlegende Idee ist hierbei die Spezifikation von Beziehungen zwischen Quell- und Zielelementtypen auf Basis von Bedingungen (engl. constraints). Relationale Ansätze unterstützen grundsätzlich bidirektionale Regeln und verlangen meist eine strikte Trennung von Quell- und Zielmodellen, d.h. sie erlauben keine Modellmodifikation. Ein Vertreter dieser Katagorie ist das *Model Transformation Framework*[36] (MTF) der *International Business Machines Corporation* (IBM) [4] [26].

Wie der Name bereits aussagt, bedienen sich auf Graphtransformationen basierende Verfahren der theoretischen Grundlagen der Graphtransformation [4] [26]. Dies bedeutet aber nicht zwingenderweise, dass die betreffenden Transformationen tatsächlich in visueller Art und Weise spezifiziert werden. Modellelemente und deren Beziehungen zueinander werden hierbei als Knoten und Kanten von Graphen angesehen. Die Transformationsregeln besitzen

---

[32] Details z. B. unter https://support.sas.com/documentation/onlinedoc/91pdf/sasdoc_91/base_javameta_7655.pdf
[33] Details z. B. unter http://www.omg.org/mda/mda_files/MDA_OptimalJ.pdf
[34] Details z. B. unter http://www.kermeta.org/
[35] Details z. B. unter http://www.omg.org/spec/OCL/
[36] Details z. B. unter http://www.ibm.com/developerworks/rational/library/05/503_sebas/

jeweils eine linke (engl. left-hand side, LHS) und eine rechte Seite (engl. right-hand side, RHS) [28]. In LHS und RHS werden sogenannte Muster definiert. Das LHS-Muster wird im Quellmodell gesucht und gegebenenfalls durch das RHS-Muster ersetzt. Die LHS verfügt neben dem Muster auch oft noch über Bedingungen, die vor der Ersetzung des Musters ausgewertet werden. Zudem wird gegebenenfalls weitere Logik für die Berechnung von Zielattributwerten, z. B. von Elementnamen, benötigt [4] [26]. Ähnlich wie die relationalen Verfahren sind auch die Graphtransformationen dazu geeignet, die Modelltransformationen in deklarativer Weise zu formulieren [26]. Zu dieser Gruppe der M2M-Transformationen gehört beispielsweise das Werkzeug *Graph Rewriting and Transformation*[37] (GReAT) [4] [26].

Hybride Verfahren kombinieren Techniken aus den bisher beschriebenen M2M-Transformationsansätzen [4] [26]. Die Kombination kann hierbei auf Basis ganzer Verfahren als auch auf der niedrigeren Ebene der einzelnen Transformationsregeln erfolgen [4]. Beispiel für einen hybriden Transformationsansatz ist die *ATLAS Transformation Language* (ATL), die sowohl imperative als auch deklarative Elemente unterstützt [27]. Details zu ATL sind beispielswiese in [30] zu finden.

In der Kategorie der sonstigen Transformationsverfahren fällt die *Extensible Stylesheet Language Transformation*[38] (XSLT) [4] [26]. Grundsätzlich besteht die Möglichkeit, Modelle mittels *XML Metadata Exchange*[39] (XMI) in serielle Form konform zur *Extensible Markup Language*[40] (XML) zu bringen. Die so erzeugten XML-Dokumente können dann anschließend unter Zuhilfenahme von XSLT transformiert werden [4] [26]. XSLT ist die Standardtechnologie zur Transformation von XML-Dokumenten. Dieser Transformationsansatz kann als Umformulierung von Ausdrücken mittels einer funktionalen Sprache verstanden werden. Es ergeben sich jedoch aufgrund der Verbosität und schlechten Lesbarkeit Probleme hinsichtlich der Skalierbarkeit [4].

2.3.2.2. Modell-zu-Text-Transformationen

Wie bereits aus dem Namen ersichtlich, handelt es sich bei der Modell-zu-Text-Transformation (engl. model-to-text, M2T) um die Umformung eines Modells in eine textuelle Darstellung bzw. in Zeichenketten. Insbesondere im Kontext der modellgetriebenen Software- bzw. UI-Entwicklung wird hierfür häufig auch die Bezeichnung Modell-zu-Code-Transformation (engl. model-to-code, M2C) verwendet und meint die abschließende Generierung von Source-Code aus einem PSM (MDA) oder eines FUI aus einem CUI (CRF), z. B. in [26]. Grundsätzlich könnte eine solche Modell-zu-Code- auch als Modell-zu-Modell-Transformation angesehen werden. Der Code wäre dann als Abstraktion des betreffenden Maschinencodes zu interpretieren, was durchaus nachvollziehbar erscheint [4]. Bei einer Transformation zwischen Modellen müsste diese, wie in Abbildung 7 gezeigt, das Metamodell der jeweiligen Zielprogrammiersprache mit einbeziehen. Aus praktischen Gründen wird aber meist nur einfacher Text erzeugt, der dann einem Copiler der Zielplattform zugeführt wird [26]. Zudem besteht die Möglichkeit, dass ausdrücklich kein

[37] Details z. B. unter http://www.isis.vanderbilt.edu/tools/great
[38] Details z. B. unter http://www.w3.org/TR/xslt
[39] Details z. B. unter http://www.omg.org/spec/XMI/
[40] Details z. B. unter http://www.w3.org/XML/

Code, sondern beispielsweise XML-Artefakte oder Text zu Dokumentationszwecken generiert werden soll. Aus diesen Gründen wird im Folgenden der allgemeinere Begriff der Modell-zu-Text-Transformation verwendet.

In [4] und [26] wird hinsichtlich der M2T-Transformationen zwischen den sogenannten besucherbasierten (engl. visitor-based) und schablonenbasierten (engl. template-based) Verfahren unterschieden.

Bei einem besucherbasierten Verfahren handelt es sich um einen sehr einfachen Transformationsansatz [4] [26], der auf dem in [31] beschriebenen *Visitor* Pattern basiert. Die interne Repräsentation des Quellmodells wird dabei gemäß einer festgelegten Strategie, z. B. mit der Tiefensuche (engl. depth-first), durchlaufen [32] und erzeugt schrittweise einen Text-Strom aus den Transformationsergebnissen [4] [26]. Ein solches Verfahren ist vergleichsweise einfach zu implementieren, orientiert sich aber stark an der Struktur des Quellmodells. Besucherbasierte Textgeneratoren sind in der Praxis relativ selten anzutreffen [32]. Ein typisches Anwendungsbeispiel ist der Generator *Jamda*[41] [4] [26] [32].

Die Mehrheit der verfügbaren MDA-Werkzeuge unterstützt schablonenbasierte M2T-Transformationen. Diese basieren auf den gleichen theoretischen Grundlagen wie die schablonenbasierten M2M-Transformationen (siehe Kapitel 2.3.2.1). Eine Schablone besteht hierbei aber aus Zieltexten mit eingebettetem Metacode, der zum Zugriff auf das Quellmodell und zur Durchführung von Selektionen und iterativen Erweiterungen dient [4] [26]. Die Abfrage von Elementen des Quellmodells kann entweder imperativ oder deklarativ erfolgen. Darüber hinaus ist für alle Schablonen festgelegt, auf welche Elemente des Quellmodells sie anwendbar sind. Bei der Transformation wird der Text unmittelbar ausgegeben, wobei der enthaltene Metacode gemäß seiner jeweiligen Bedeutung aufgelöst wird [32]. Im Gegensatz zu einem besucherbasierten Ansatz ähnelt die Struktur einer Schablone dem zu generierenden Text [4] [26]. Zusammengehörende Teile der Zielsprache können in einer Schablone zusammengefasst werden, wodurch sich die Lesbarkeit der Textgenerator-Spezifikation deutlich erhöht [32]. Schablonenbasierte Transformationen werden beispielsweise bei *AndroMDA*[42] eingesetzt [4] [26].

2.3.2.3. Text-zu-Modell-Transformationen

Bei Text-zu-Modell-Transformationen (engl. text-to-model, T2M) wird aus Text eine Modellbeschreibung erzeugt. Es handelt sich um praktisch die Umkehrung von M2T-Transformationen. Hierbei kommen Parsing- und Reverse-Engineering-Techniken zum Einsatz [4].

2.3.3. Transformationsansätze

Neben den bereits in Kapitel 2.3.2 genannten gibt es noch eine Vielzahl weiterer existierender Transformationsansätze, zu denen reichhaltige Literatur mit Beschreibungen, Bewertungen und Vergleichen verfügbar sind, z. B. in [4] [6] [26] [27] [33].

---

[41] Siehe http://jamda.sourceforge.net/
[42] Siehe http://www.andromda.org/

An dieser Stelle möchte ich noch einen Ansatz etwas detaillierter beschreiben, da hierbei die Zusammenhänge und generelle Funktionsweisen von Transformationen gut ersichtlich sind. Es handelt sich dabei um die durch die *Object Management Group* (OMG) standardisierte Sprache für die Spezifikation von Abfragen, Sichten und Transformationen (engl. *Query, Views, Transformations*[43], QVT) [34]. Gemäß der Terminologie dieser OMG-Spezifikation dienen hierbei Abfragen dazu, Modellelemente aufzufinden und auszuwählen, die anschließend transformiert werden sollen. Als Sichten werden Modelle bezeichnet, die von anderen Modellen abgeleitet werden. Transformationen sind schließlich die Abbildungen von Elementen des Quell- auf Elemente des Zielmodells [35]. QVT bezieht sich auf Modelle, die auf dem ebenfalls durch die OMG definierten Standard *MetaObject Facility* [44] (MOF) basieren. MOF steht in direktem Zusammenhang mit dem Metamodellierungsansatz der OMG, definiert eine abstrakte Sprache zur Beschreibung von Modellierungssprachen und kann somit als Metametamodell interpretiert werden. Die Rolle von QVT in diesem Kontext wird in Abbildung 8 illustriert [34].

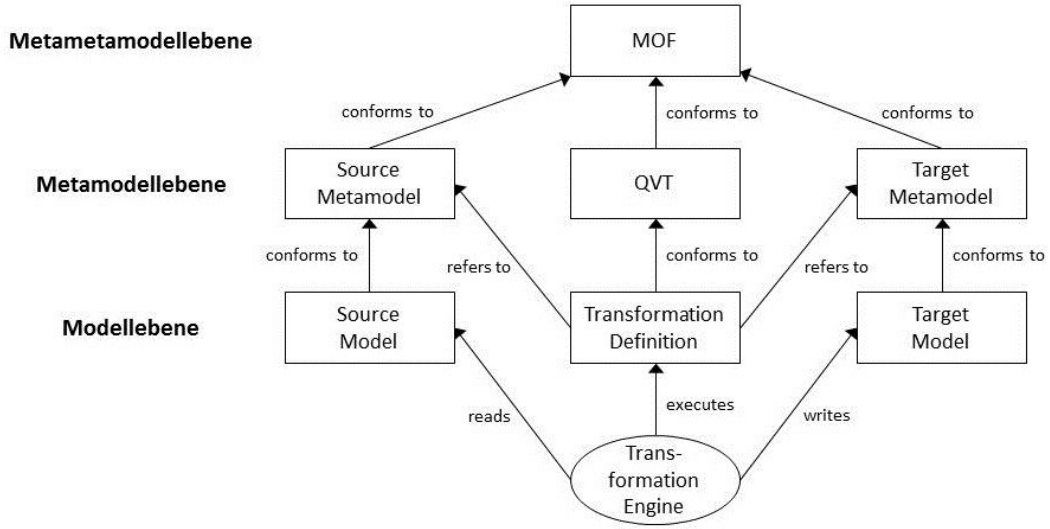


Abbildung 8 ROLLE VON QVT IM TRANSFORMATIONSKONTEXT [34]

Die abstrakte Syntax der Sprache ist als MOF-konformes Metamodell spezifiziert. Wie auch schon in Abbildung 7 gezeigt, basieren die Transformationen auf den Quell- und Zielmetamodellen. Die Ausführung der Transformationen erfolgt auf Instanzen dieser Metamodelle, also auf den Quell- und Zielmodellen. Die grundlegende Architektur von QVT ist in Abbildung 9 dargestellt [34].

QVT besteht aus deklarativen Anteilen, die in den zwei Ebenen *Relations* und *Core* strukturiert sind, und den imperativen Teilen *Operational Mappings* und *Black Box* [36]. Mit QVT sind drei Sprachen für M2M-Transformationen definiert. *QVT Relational* ist eine deklarative Transformationssprache höherer Ebene mit sowohl einer graphischen als auch mit einer textuellen Syntax. Sie unterstützt die Spezifikation bidirektionaler Transformationen, die als

[43] Engl.: Abfragen, Sichten und Transformationen
[44] Siehe auch http://www.omg.org/spec/MOF/

Menge von Relationen zwischen den Quell- und Zielmetamodellen definiert sind und erfüllt sein müssen. *QVT Core* ist eine einfache, deklarative Modelltransformationssprache auf niedrigerer Ebene. Sie dient als Grundlage für *QVT Relational* [37]. *QVT Relational* und *QVT Core* verkörpern die gleiche Semantik, aber auf unterschiedlichen Abstraktionsniveaus. *QVT Operational* ist eine imperative Transformationssprache, die den Aufruf von Implementierungen imperativer Transformationen aus den deklarativen QVT-Sprachen heraus ermöglicht [36]. Mittels der *Black Box* können schließlich noch externe Programme während der Ausführung einer Transformation aufgerufen werden [34]. Aufgrund der Existenz von deklarativen als auch imperativen Anteilen ist QVT als hybrider Transformationsanatz einzuordnen [4].

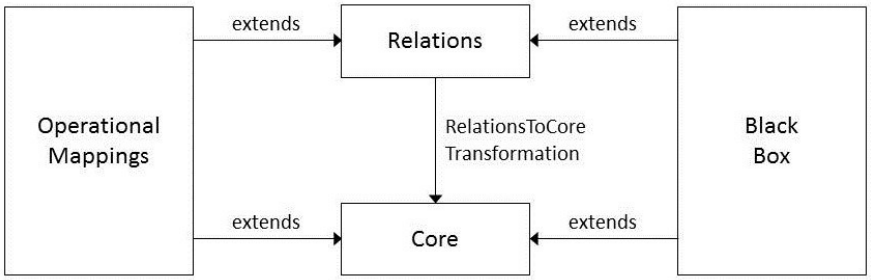


Abbildung 9 GRUNDLEGENDE ARCHITEKTUR VON QVT [34]

2.4. Modellierungssprachen

Das folgende Kapitel widmet sich der Möglichkeiten der formalen Beschreibung von Benutzerschnittstellen mittels *User Interface Description Languages*[45] (UIDL). Eine UIDL ist eine Sprache hohen Abstraktionsgrades zur Beschreibung der hinsichtlich der betreffenden interaktiven Anwendung relevanten Eigenschaften einer Benutzeroberfläche. Für solch eine Sprache muss sowohl deren Syntax, als auch die Semantik definiert sein. Die Syntax legt fest, wie die Eigenschaften mithilfe der UIDL ausgedrückt werden und die Semantik spezifiziert deren Bedeutung in der Realität. Eine UIDL dient dazu, die Benutzerschnittstelle unabhängig von den Zielprogrammiersprachen, mit denen sie schließlich implementiert werden soll, zu beschreiben [38].

Im Rahmen des Dissertationsprojekts wurden Literaturreviews hinsichtlich existierender UIDL durchgeführt. Bei deren Auswahl wurde darauf geachtet, dass jeweils die Möglichkeit besteht, die Benutzerschnittstellen in Konformität zur *Extensible Markup Language*[46] (XML) zu beschreiben. In Unterkapitel 2.4.1 werden folgende UIDL kurz beschrieben und erklärt: *User Interface Markup Language* (UIML), *XML User Interface Language* (XUL), *Extensible Interface Markup Language* (XIML), *Interface Specification Meta-Language* (ISML), *User Interface Extensible Markup Language* (UsiXML), *Dialog Modeling Language* (DiaMODL), *Transformation Environment for Interactive Systems Representations* (TERESA XML) und

[45] Engl.: Benutzerschnittstellen-Beschreibungssprachen
[46] Engl.: erweiterbare Auszeichnungssprache

Model-based Language for Interactive Applications (MARIA XML). Abbildung 10 gibt einen Überblick über die zeitliche Einordnung der untersuchten Modellierungssprachen.

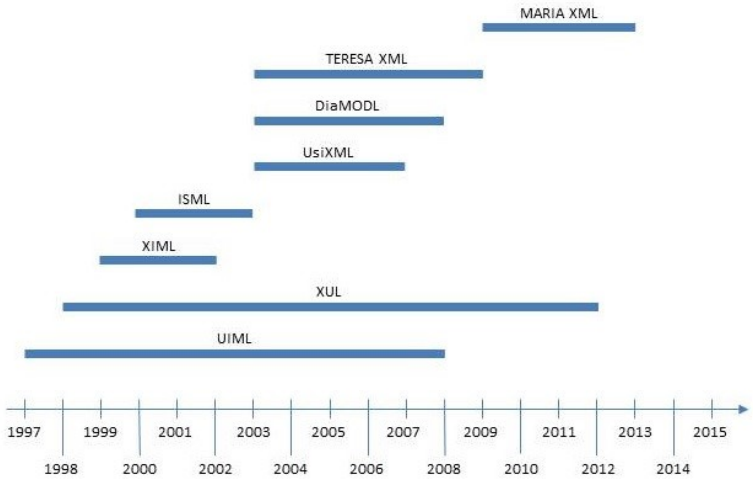


Abbildung 10 ZEITLICHE EINORDNUNG DER UNTERSUCHTEN UIDL

Anschließend erfolgen in Kapitel 2.4.2 ein Vergleich und die Bewertung der Sprachen anhand eines zuvor festgelegten Schemas. Insbesondere wurde untersucht, welche Abstraktionsgrade des CAMELEON Reference Framework (siehe Kapitel 2.1.3) durch die jeweilige Sprache abgedeckt werden können. Die betreffenden Inhalte wurden bereits in verkürzter Form in [39] veröffentlicht.

2.4.1. Untersuchte UI-Beschreibungssprachen

2.4.1.1. UIML

Die *User Interface Markup Language*[47] (UIML) ist eine gemeinsame Entwicklung der *Virginia Polytechnic Institute and State University*[48] (USA) und der Firma *Harmonia Inc*. (USA). Der ursprüngliche Entwurf der Sprache geht auf das Jahr 1997 zurück [40]. Die erste Version, UIML 1.0, wurde 1998 freigegeben. Die jüngste Fassung ist UIML 4.0, die seit 2008 als sogenannter *Committee Draft* der *Organization for the Advancement of Structured Information Standards*[49] (OASIS) vorliegt [41].

Auf oberster Ebene besteht ein UIML-Dokument aus zwei Teilen, einem Prolog (Prologue) und einem Wurzel-Element (Root). Im Prolog, mit dem das Dokument beginnt, sind die Versionsnummer der verwendeten XML-Sprache, die Kodierung (*Encoding*) und der Verweis auf die UIML *Document Type Definition* (DTD) hinterlegt. Das *Root*-Element, das mit dem *Tag* <uiml> gekennzeichnet wird, umfasst vier weitere, optionale Elemente. Dies sind Kopf (*Head*), Vorlage (*Template*), Schnittstelle (*Interface*) und das *Peers*-Element. Das *Head*-

[47] Engl.: Auszeichnungssprache für Benutzerschnittstellen
[48] Polytechnisches Institut und Universität des US-Bundesstaates Virginia
[49] Engl.: Organisation zur Weiterentwicklung strukturierter Informationsstandards

Element beinhaltet Metadaten bezüglich des vorliegenden UIML-Dokuments, die weder Teil der zu beschreibenden Benutzeroberfläche sind, noch deren Umsetzung oder Funktionsweise beeinflusst. Das *Template*-Element ermöglicht Wiederverwendbarkeit, indem es UIML-Code spezifiziert, der in anderen Abschnitten des Dokuments benutzt werden kann. Das *Peers*-Element ermöglicht Erweiterbarkeit, indem es *Mappings* von Namen von Klassen (*Classes*), Eigenschaften (*Properties*), Ereignissen (*Events*) und Aufrufen (*Calls*) zu Entitäten außerhalb des UIML-Dokuments definiert. Das *Interface*-Element enthält schließlich jegliche Information hinsichtlich der Repräsentation der Benutzerschnittstelle [41]. Es hat folgende vier Hauptbestandteile: Struktur (*Structure*), Stil (*Style*), Inhalt (*Content*) und Verhalten (*Behavior*). Hierbei beschreibt das *Structure*-Element den physikalischen Aufbau der Benutzeroberfläche einschließlich der Beziehungen zwischen deren Elementen. Innerhalb des *Style*-Elements werden diejenigen Eigenschaften und Werte spezifiziert, die bei der Umsetzung der Benutzerschnittstelle benötigt werden, beispielsweise also Hintergrundfarbe, Schriftart und Schriftgröße. Mit dem *Content*-Element wird eine strikte Trennung zwischen Inhalt und Struktur der Oberfläche erreicht. Es findet die Zuordnung der eigentlichen Inhalte zu den Elementen der Schnittstelle statt. Das *Behavior*-Element legt abschließend das Verhalten der Benutzeroberfläche fest. Dies wird durch Regeln (*Rules*) erreicht, die ihrerseits aus Bedingungen (*Conditions*) und zugeordneten Aktionen (*Actions*) bestehen. UIML unterscheidet zwischen folgenden vier Aktionstypen [40]:

1. Zuweisung eines Wertes zu einer Eigenschaft eines Schnittstellenelements
2. Aufruf einer externen Funktion oder Methode
3. Auslösen eines Ereignisses
4. Umstrukturierung der Benutzeroberfläche

UIML unterstützt die Entwicklung von Multi-Plattform-Benutzerschnittstellen. Das *Structure*-Element muss jedoch für jedes Endgerät beziehungsweise jede Plattform erneut spezifiziert werden. Das bedeutet, dass die Benutzeroberfläche für die verschiedenen Plattformen zwar mit einer einzigen Sprache beschrieben werden können, aber das eigentliche Design weiterhin wiederholt stattfinden muss [40]. Hinsichtlich des Abstraktionsgrades deckt UIML die Modell-Ebene ab.

2.4.1.2. XUL

Im Jahr 1998 hat die *Netscape Communications Corporation* (USA) den Quellcode ihres Internet-Browsers veröffentlicht und das Open-Source-Projekt *Mozilla* zu dessen Weiterentwicklung ins Leben gerufen. Im Zuge dieses Übergangs wurde auch die *XML User Interface Language*[50] (XUL) eingeführt, um die Entwicklung der Benutzeroberflächen der Mozilla-Produkte zu vereinheitlichen und zu erleichtern [42].

Mit XUL können Inhalte mit bestimmtem Verhalten und die Präsentation dieser Inhalte beschrieben werden. Ferner werden Möglichkeiten hinsichtlich der Software-Lokalisierung[51] (*Localization*) angeboten. Wenn plattformübergreifende Web-Applikationen unter

[50] Auf der *Extensible Markup Language* (XML) basierende Benutzerschnittstellen-Sprache
[51] Anpassung an örtliche Gegebenheiten, insbesondere der Sprache

Zuhilfenahme von XUL entwickelt werden, so lassen sich auch andere Technologien einsetzen, wie zum Beispiel *Cascading Style Sheets*[52] (CSS), *Extensible Binding Language*[53] (XBL), die ein Hilfsmittel zum Ändern, Ersetzen oder Hinzufügen von XML-*Tags* darstellt, *Overlays*[54], *Cross Platform Component Object Model* (XPCom)[55] / *XPConnect*[56] zur Integration neuer Bibliotheken (*Libraries*) und *Cross Platform Install* (XPInstall), das die Installation von XUL-Anwendungen aus dem Internet oder Intranet erlaubt [43].

Während native XUL-Elemente dazu dienen, das Layout einer Benutzeroberfläche zu beschreiben, wird das Verhalten durch den Einsatz einer Skriptsprache, meist JavaScript[57], erzeugt. Ein grundlegendes Konzept von XUL ist das *Box*-Modell. Hierbei wird jedes XUL-Element zunächst als rechteckige Box, die wiederum weitere Boxen beinhalten kann, angesehen. Es wird zwischen horizontalen Boxen (hbox), bei denen die enthaltenen Boxen nebeneinander, und vertikalen Boxen (vbox), bei denen die betreffenden Boxen untereinander angeordnet sind, unterschieden [42]. Letztendlich wird die Benutzeroberfläche durch eine Menge von strukturierten UI-Elementen, wie Window, Menu Bar, Button, Checkbox und dergleichen, repräsentiert [44].

Generell ist XUL auf allen Betriebssystem-Plattformen verfügbar, auf denen die *Mozilla Rendering Engine* namens *Gecko* läuft. Dazu zählen BSD[58] Unix, Linux, OS X[59], Solaris[60], OS/2[61], AIX[62], OpenVMS[63] und Windows[64] [45]. Hinsichtlich des Abstraktionsgrades deckt XUL die CUI-Ebene ab.

2.4.1.3. XIML

Die *Extensible Interface Markup Language*[65] (XIML) wurde von der Firma *RedWhale Software* (USA) entwickelt [46]. Die erste Version, XIML 1.0, wurde im Jahr 1999 veröffentlicht [47]. Das grundlegende Prinzip ist die strenge Trennung der Definition der Benutzerschnittstelle und deren eigentliches Aussehen auf der Zielplattform [48]. Mit XIML können sowohl abstrakte als auch konkrete UI-Aspekte beschrieben werden, also beispielsweise der Kontext der Benutzerinteraktionen oder auch bestimmte Widgets[66], die auf dem Bildschirm dargestellt werden sollen [46].

---

[52] Ergänzungssprache zur *Hyper Text Markup Language* [205] (HTML), welche die Formatierung und exakte Positionierung von HTML-Elementen erlaubt

[53] XML-basierte Auszeichnungssprache, die Verhalten und Aussehen von XML- und HTML-Elementen beschreibt [206]

[54] XUL-Dateien, die zusätzliche UI-Inhalte beschreiben [44]

[55] Plattformunabhängiges Komponenten-Modell von Mozilla basierend auf Schnittstellenbeschreibungen [68]

[56] Möglichkeit des Aufrufs von XPCom in Skripten [44]

[57] JavaScript ist eine Programmiersprache, die in HTML-Dateien eingebunden werden kann, ohne jedoch direkter Bestandteil von HTML zu sein [205]

[58] Berkely Software Distribution

[59] Auf Unix basierendes 64-Bit-Betriebssystem für *Macintosh*-Rechner der Firma *Apple Inc.* [207]

[60] Auf Unix basierendes Server-Betriebssystem der *Oracle Corp.* [209]

[61] Von der *IBM Corp.* zusammen mit der *Microsoft Corp.* entwickeltes Betriebssystem, für das der Support im Jahr 2006 eingestellt wurde [208]

[62] Von der *International Business Machines Corp.* (IBM) entwickeltes Unix-Betriebssystem [210]

[63] Ursprünglich von der *Digital Equipment Corp.* (DEC) entwickeltes Betriebssystem, das mittlerweile der *Hewlett-Packard Development Company, L.P.* (HP) gehört [211]

[64] Von der *Microsoft Corp.* entwickeltes Betriebssystem [212]

[65] Engl.: Erweiterbare Schnittstellen-Auszeichnungssprache

[66] UI-Objekte einer bestimmten Zielplattform

XIML kann als eine organisierte Sammlung von Benutzerschnittstellenelementen angesehen werden, die wiederum in UI-Komponenten kategorisiert sind. Theoretisch ist die Anzahl der Typen und Komponenten nicht begrenzt, aber XIML 1.0 umfasst fünf vordefinierte, grundlegende Komponenten: Aufgaben- (*Task*), Domänen- (*Domain*), Benutzer- (*User*), Präsentations- (*Presentation*) und Dialog-Komponenten. Während die Aufgaben-Komponente den Geschäftsprozess und die durch die Schnittstelle unterstützten Aufgaben des Benutzers bestimmt, spezifiziert die Domänen-Komponente die Klassen und Datenobjekte, die dem Anwender angezeigt oder durch diesen manipuliert werden. Die Benutzer-Komponente dient dazu, Eigenschaften einzelner Anwender und ganzer Benutzergruppen festzulegen und die Präsentations-Komponente definiert eine Hierarchie von Interaktionselementen, die die konkreten Objekte, die mit dem Benutzer kommunizieren, umfasst. Die Dialog-Komponente bestimmt die möglichen Interaktionen und die Navigation [46].

Darüber hinaus stellt XIML Attribute bereit, die Merkmale und Eigenschaften von Elementen repräsentieren und denen Werte zugewiesen werden können. Zudem gibt es die Möglichkeit, Relationen zwischen zwei oder mehreren XIML-Elementen zu definieren [46].

XIML deckt sowohl die Abstraktionsgrade der Modell- [46] als auch der abstrakten (AUI) und konkreten (CUI) Ebene ab [48].

2.4.1.4. ISML

Die *Interface Specification Meta Language*[67] (ISML) wurde im Jahr 2003 an der *Bournemouth University* (UK) entwickelt. Sie gehört zu einem modellbasierten Benutzerschnittstellen-Framework, das auf Metapher-Modellen basiert. Die Metaphern werden dazu benutzt, um zwischen Benutzer und System zu vermitteln. Das Ziel von ISML ist, die Metaphern zwar im Design explizit umzusetzen, jedoch die Metapher-Modelle streng von der jeweiligen Implementierung zu trennen. Grundsätzlich kombiniert das Framework Aufgaben-Modelle und Interaktor-Definitionen und sieht auf Metaphern basierende Zuordnungen zwischen diesen vor [49].

Das ISML Framework besteht aus fünf Teilen: Geräte (*Devices*), Komponenten (*Components*), Meta-Objekten (*Meta Objects*), Interaktoren (*Interactors*) und Aufgaben (*Tasks*). Hierbei bezeichnen die Geräte einfache Abstraktionen von Benutzer-Ein- und Ausgabe-Hardware, wie beispielsweise Tastatur und Maus. Komponenten sind logische Abstraktionen von Benutzer-Ein- und Ausgabe-Objekten, durch die sich die Funktionalitäten der Geräte beschreiben lassen. Meta-Objekten kommt eine Schlüsselrolle sowohl hinsichtlich der Interaktions- als auch der Aufgaben-Domäne zu. Hierdurch erzeugte Objekte besitzen Eigenschaften (*Attributes*), Zustände (*States*), Bedingungen (*Constraints*) und Kommunikationsmechanismen (*Communication Mechanisms*). Die Ebene der Meta-Objekte bildet die Grundlage für die Spezifikation der Metapher-Abstraktions-Schicht, deren Implementierung in der Benutzerschnittstelle in Form von Interaktoren und das Aufgaben-Modell. Interaktor-Definitionen basieren auf Meta-Objekten und stellen eine bestimmte Design-Lösung dar [50]. Die Aufgaben-Schicht kombiniert Meta-Objekt-Definitionen von

[67] Engl.: Metasprache zur Schnittstellenspezifikation

Objekten und Aktionen mit einer einfachen, hierarchischen Dekomposition der Aufgaben und beschreibt, wie die Benutzerziele letztlich erreicht werden [49].

ISML unterstützt Entwurf und Entwicklung von Benutzeroberflächen für unterschiedliche Plattformen [49]. Es wird der Abstraktionsgrad der Modell-Ebene abgedeckt.

2.4.1.5. UsiXML

Die *User Interface Extensible Markup Language*[68] (UsiXML) wurde von der *Information Systems Research Unit* (ISYS) der *Université Catholique de Louvain*[69] (UCL, Belgien) entwickelt [51] und im Jahr 2003 veröffentlicht [52]. Die Sprache rührt vom CAMELEON FP5-Projekt her [53] und hält deshalb die vier Abstraktionsebenen des *CAMELEON Reference Framework* (siehe Kapitel 2.1.3) ein: Aufgaben & Konzepte (*Task & Concepts*), abstrakte Benutzerschnittstelle (*Abstract User Interface*), konkrete Benutzerschnittstelle (*Concrete User Interface*) und finale Benutzerschnittstelle (*Final User Interface*) [51]. Die aktuelle Version UsiXML 1.8 stammt aus dem Jahr 2007 [54].

UsiXML schließt mehrere Modelle zum Entwurf von Benutzerschnittstellen ein: Aufgaben-Modell (*Task Model*), Domänen-Modell (*Domain Model*), Präsentations-Modell (*Presentation Model*), Dialog-Modell (*Dialog Model*) und Benutzungskontext-Modell (*Context of Use Model*). Letzteres setzt sich aus Benutzer-Modell (*User Model*), Plattform-Modell (*Platform Model*) und Umgebungs-Modell (*Environment Model*) zusammen [51]. Beziehungen zwischen den Modellen werden in *Mapping*-Modellen definiert und dokumentiert. Darüber hinaus unterstützt UsiXML noch Transformations-Modelle, in denen Transformationen mithilfe von sogenannten Transformations-Systemen, die aus der Grafentheorie entstammen, spezifiziert werden [55].

UsiXML erlaubt es, eine Benutzeroberfläche auf allen vier Abstraktionsebenen des CAMELEON Reference Frameworks zu spezifizieren [55] und unterstützt die Definition der Schnittstelle unabhängig von Endgerät, Plattform, Modalität und Benutzungskontext [51]. Für UsiXML sind zahlreiche Werkzeuge verfügbar und zudem sind vielfältige Zielsprachen möglich (siehe Anhang A, Tabelle 42). Darüber hinaus werden diverse Plattformen unterstützt, wie zum Beispiel Mobiltelefon, PocketPC, *Interactive Kiosk*, *Wall Screen* und *Personal Digital Assistant* (PDA) [52].

2.4.1.6. DiaMODL

Die *Dialog Modeling Language*[70] (DiaMODL) wurde von der *Norwegian University of Science and Technology*[71] (NTNU, Norwegen) entwickelt und eingeführt [56]. Die erste Veröffentlichung fand im Jahr 2003 statt. DiaMODL ist eine hybride Beschreibungssprache, die Interaktoren[72] (*Pisa Interactor Abstraction*) und *UML State Charts*[73] kombiniert [57].

---

[68] Engl.: Erweiterbare Auszeichnungssprache für Benutzerschnittstellen
[69] Franz.: Katholische Universität Louvain
[70] Engl.: Dialogmodellierungssprache
[71] Engl.: Norwegische Universität der Wissenschaft und Technologie
[72] Pisa Interactor Abstraction [213]
[73] Zustandsdiagramme der Unified Modeling Language (siehe beispielsweise [214])

Die Interaktoren spielen die Rolle von Informationsvermittlern zwischen einer Komponente und dem interaktiven System, indem sie Informationen mittels sogenannter Gatter (Gates) senden und empfangen. Die Gatter werden als Dreiecke visualisiert und bestehen jeweils aus einer Spitze (Tip) und einer Grundlinie beziehungsweise Basis (Base).

Es gibt vier grundsätzliche Arten von Gattern:

1. Input/Send (Eingabe/Senden)
   Benutzereingaben haben zur Folge, dass Informationen zum interaktiven System gesendet werden.
2. Output/Receive (Ausgabe/Empfangen)
   Systemausgaben werden empfangen und daraufhin Informationen an den Benutzer gesendet.
3. Input/Receive (Eingabe/Empfangen)
   Benutzereingaben werden vom Interaktor zur weiteren Verarbeitung oder Vermittlung empfangen.
4. Output/Send (Ausgabe/senden)
   Ausgabe-Informationen werden zum Benutzer gesendet.

Ein Wert, der an der Basis eines Gatters empfangen wird, wird mithilfe einer Funktion verarbeitet und anschließend über die Spitze weitergeleitet. Durch Verbinden von Gattern können ganze Netzwerke von Interaktoren gebildet werden. Ähnlich wie die Gatter, können auch die Verbindungen mit Funktionen verknüpft werden. Um diese zu beschreiben, wird typischerweise eine *Domain Modeling Language*[74] verwendet, beispielsweise UML. Interaktoren werden hauptsächlich verwendet, um *Concrete Interaction Objects*[75] (CIO) zu definieren [57].

Die dynamischen Aspekte der Benutzerschnittstelle, wie beispielsweise das Auslösen von Informationsflüssen und die Aktivierung oder Deaktivierung von Interaktoren, werden mittels Zustandsdiagrammen modelliert. Hierfür wurde das Meta-Modell von UML entsprechend erweitert [57].

DiaMODL unterstützt den Entwurf und die Entwicklung sowohl von Benutzerschnittstellen für verschiedene Plattformen, als auch von unterschiedlichen Interaktionsformen [57]. Hinsichtlich des Abstraktionsgrades wird die Modell-Ebene abgedeckt.

2.4.1.7. TERESA XML

Die Sprache TERESA XML ist integraler Bestandteil des *Transformation Environment for Interactive Systems Representations*[76] (TERESA) [58], das von der *Human-Computer Interaction* (HCI) Gruppe des italienischen *Istituto di Scienza e Tecnologie dell' Informazione*[77] (ISTI) entwickelt wurde. ISTI wiederum ist ein Institut des *Consiglio Nazionale*

[74] Engl.: Domänenmodellierungssprache
[75] Engl.: Konkrete Interaktionsobjekte
[76] Engl.: Transformationsumgebung für interaktive Systemrepräsentationen
[77] Ital.: Institut für Informationswissenschaft und -technologie

delle Ricerche[78] (CNR) [59]. Es unterstützt den Entwurf und die Entwicklung von Multi-Device-Benutzerschnittstellen. Die Arbeiten an TERESA begannen im Jahr 2003 [60].

TERESA bedient sich einer Methode des modellbasierten Designs. Zunächst werden ein High-Level-Task-Modell und ein Domänen-Modell, das alle für die Ausführung der Aufgaben benötigten Interaktionsobjekte umfasst, definiert. Anschließend werden sogenannte System-Task-Modelle ausgearbeitet, die praktisch als plattformspezifische Verfeinerungen und Anpassungen des originalen Aufgaben-Modells angesehen werden können. Aus diesen werden dann abstrakte Schnittstellenbeschreibungen gewonnen. Sie bestehen aus Präsentationen beziehungsweise Interaktoren und den jeweiligen Beziehungen zwischen diesen. Hierbei definieren die Präsentationen den statischen Aufbau der Benutzeroberfläche, während deren Beziehungen das dynamische Verhalten spezifizieren. Die Verbindungen zwischen den Präsentationen können auch als eine Art Dialog-Modell interpretiert werden. Durch Ersetzen der Interaktoren in den AUIs durch konkrete Interaktionsobjekte der Zielplattform werden die CUIs erzeugt. Diese bilden schließlich die Grundlage für die Generierung der endgültigen Benutzeroberfläche in der gewünschten Zielsprache, wie beispielsweise XHTML[79] oder Java[80] [61], [62].

Generell gesehen besteht TERESA XML aus zwei Teilen. Einerseits wird das Aufgabe-Modell in *ConcurTaskTrees* (CTT) Notation [63] repräsentiert und wird in einem entsprechenden, XML-konformen Format gespeichert. Andererseits umfasst es die oben genannten abstrakten und konkreten Beschreibungen in Form von AUI und CUI [61], [62].

Hinsichtlich des Abstraktionsgrades deckt TERESA XML die Modellebene sowie AUI und CUI ab.

2.4.1.8. MARIA XML

Die *Model-based Language for Interactive Applications*[81] (MARIA) ist eine, ebenfalls XML-konforme, Weiterentwicklung von TERESA (siehe Kapitel 2.5.1.16) und wurde ebenfalls von der HCI Gruppe des ISTI-CNR entwickelt [60]. Die ersten Veröffentlichungen datieren aus dem Jahr 2009 [64], [65].

Auf der Basis der mit TERESA gemachten Erfahrungen und einer Analyse des aktuellen Stands der Technik wurden zusätzliche Anforderungen an eine moderne UIDL identifiziert [64]:

1. Mehr Einflussnahme durch den Designer auf die erzeugte Benutzeroberfläche mittels eines Event-Modells
2. Flexiblere Dialog- und Navigations-Modelle, die parallele Interaktionen unterstützen
3. Ein flexibleres Daten-Modell, das die Verbindung verschiedener Datentypen an die unterschiedlichen Interaktoren erlaubt
4. Unterstützung neuester dynamischer Technologien, wie beispielsweise AJAX

---

[78] Ital.: Nationaler Forschungsrat

[79] Extensible Hyper Text Markup Language, standardisiert durch das *W3C*, siehe http://www.w3.org/TR/xhtml1/

[80] Objektorientierte Programmiersprache; entwickelt von der seit 2010 zur *Oracle Corporation* gehörenden Firma *Sun Microsystems*

[81] Engl.: Modellbasierte Sprache für interactive Anwendungen

5. Verschlankung der AUI- und CUI-Spezifikationen, um deren Größe zu reduzieren und Lesbarkeit zu erhöhen

In MARIA wird das Daten-Modell mithilfe der Sprache *XML Schema Definition* (XSD) beschrieben. Hinsichtlich des Event-Modells wurden zwei Arten von Ereignissen identifiziert [64]:

1. *Property Change Events*[82], die zu einer Statusänderung bestimmter Eigenschaften der Benutzeroberfläche führen
2. *Activation Events*[83], die die Aktivierung einzelner Anwendungsfunktionalitäten mittels Interaktoren erlauben

Um die kontinuierliche Aktualisierung von UI-Feldern zu ermöglichen, besitzen die Interaktoren nun ein zusätzliches boolesches Attribut, das beispielsweise dazu verwendet werden kann, asynchrone AJAX[84]-Mechanismen einzusetzen. Darüber hinaus unterstützt MARIA die dynamische Änderung von Teilen der Benutzeroberfläche. Dies betrifft sowohl die Art und Weise, wie UI-Elemente in einer Präsentation angeordnet sind, als auch die Navigation zwischen den Präsentationen [64].

Auf der AUI-Ebene wird eine Benutzerschnittstelle aus einem Daten-Modell und einer oder mehreren Präsentationen gebildet. Eine Präsentation besteht wiederum aus einem Daten- und einem Dialog-Modell, die Informationen über die Ereignisse, die durch die Präsentation ausgelöst werden können, beinhalten. Das dynamische Verhalten der Ereignisse wird anhand der temporalen Operatoren im CTT Task-Modell spezifiziert. Die CUI-Beschreibung erfolgt in Abhängigkeit zur eingesetzten Plattform, ist aber immer noch unabhängig von der Ziel-Programmsprache [64].

Bezüglich des Abstraktionsgrads deckt MARIA wie schon TERESA die Modell-, AUI und CUI-Ebenen ab [66].

2.4.2. Vergleich und Bewertung

Es wurden bereits einige UIDL-Studien und -Bewertungen durchgeführt und veröffentlicht, beispielsweise [38] und [52]. Diese sind aber bereits relativ alt [38], mangeln an exaktem Nachweis hinsichtlich der Studienergebnisse [38] oder beinhalten nicht alle der oben beschriebenen UIDL [38], [52]. Nichtsdestotrotz lieferten diese Dokumente wertvolle Informationen, insbesondere hinsichtlich der zu untersuchenden UIDL-Eigenschaften.

Alle zuvor beschriebenen UIDL wurden auf folgende Merkmale untersucht:

1. Name der UIDL
2. Urheber
3. Datum der Erstveröffentlichung
4. Aktualität beziehungsweise aktuelle Version
5. Existierende Werkzeuge, welche die Nutzung der UIDL unterstützen
6. Verfügbarkeit hinsichtlich der freien Verwendung der UIDL

---

[82] Engl.: Eigenschaft-Änderungsereignisse
[83] Engl.: Aktivierungsereignisse
[84] *Asynchronous JavaScript and XML*, Konzept zur asynchronen Datenübertragung zwischen Browser und Server

7. Inhärente Modelle beziehungsweise die Typen von UI-Aspekten, die modelliert werden können
8. Anzahl der für die UIDL spezifizierten *XML-Tags*
9. Methodologie
10. Hauptsächliche UIDL-Konzepte
11. Hauptsächlich verfolgtes Designziel der UIDL, d.h. Unterstützung mehrerer Plattformen, Benutzertypen oder Benutzungsumfelder[85]
12. Unterstützte Ziel-Programmiersprachen
13. Unterstützte Ziel-Plattformen
14. Abdeckung hinsichtlich der im CAMELEON Reference Framework (siehe Kapitel 2.1.3) definierten Abstraktionsgrade „Modell-Ebene“, *Abstract User Interface* (AUI), *Concrete User Interface* (CUI), *Final User Interface* (FUI) [12] und zusätzlich „Metamodell-Ebene“.

Ein Überblick über die im Literaturreview berücksichtigten und ausgewerteten Quellen wird in Tabelle 2 gegeben.

Tabelle 2 IM LITERATURREVIEW BERÜCKSICHTIGTE QUELLEN ZU DEN MODELLIERUNGSSPRACHEN

Modellierungssprache	Quellen
UIML	[40] [41] [52]
XUL	[38] [42] [43] [44] [45] [52] [67] [68]
XIML	[46] [47] [48] [52]
ISML	[49] [50] [52] [69]
UsiXML	[51] [52] [53] [54] [55] [70] [71]
DiaMODL	[56] [57] [72]
TERESA XML	[58] [59] [60] [61] [62] [73] [74] [75]
MARIA XML	[60] [64] [65] [66] [76] [77]

Zusammenfassend lässt sich sagen, dass alle im Literaturreview untersuchten UIDL als wertvolle Beiträge hinsichtlich des Entwurfs und der Entwicklung gerade von plattformunabhängigen Benutzerschnittstellen bezeichnet werden können.

Beginnend mit der Version 1.0 im Jahr 1998 wurde UIML zur aktuellen Version 4.0 weiterentwickelt, die seit 2008 als sogenannter *OASIS Committee Draft* verfügbar ist. Es ist seitdem aber kein weiterer Entwicklungsfortschritt an der Sprache selbst mehr feststellbar. In der Literatur wird eine umfassende Werkzeugunterstützung beschrieben, jedoch sind diese Tools aktuell im Internet nicht mehr verfügbar. Aufgrund dieser beiden Tatsachen kann begründet angenommen werden, dass die Aktivitäten hinsichtlich UIML eingestellt wurden.

Im Gegensatz dazu war die Sprache UsiXML, die im Jahr 2003 eingeführt wurde, Gegenstand eines durch die Europäische Union (EU) geförderten Projekts[86], das im März 2013 endete.

[85] Analog der Eigenschaft “Ziel” (Target) in [38]
[86] Siehe http://usixml.eu

Die jüngste Sprachspezifikation geht jedoch auf das Jahr 2007 zurück. UsiXML wurde gemäß des CAMELEON Referenz-Modells entworfen und enthält ein ausgefeiltes Modellkonzept. Zudem gibt es umfassende Werkzeugunterstützung für UsiXML.

Der hybride Modellierungsansatz in DiaMODL bringt eine Erweiterung des UML-Metamodells mit sich. Die Sprache wurde im Jahr 2003 eingeführt und die jüngste verfügbare Dokumentation datiert aus 2008 [56]. Seitdem hat offensichtlich keine Weiterentwicklung mehr stattgefunden.

ISML stellt einen sehr interessanten Ansatz zur Verwendung von Metaphern beim Entwurf von Benutzerschnittstellen dar und wurde im Jahr 2003 im Rahmen einer Doktorarbeit entwickelt. Aktuellere Dokumentation war nicht auffindbar.

Die Sprachen TERESA XML und deren Nachfolgerin MARIA XML wurden, wie auch UsiXML, gemäß des CRF entworfen und haben ihren Ursprung im Jahr 2003. Es werden die Abstraktionsebenen „Modell“, „AUI“ und „CUI“ abgedeckt. Die aktuelle Version des MARIAE Framework[87] wurde im Herbst 2013 freigegeben.

XIML ist die einzige betrachtete UIDL, die nicht nur die Entwicklung von Multi-Plattform- sondern auch von Multi-User-Benutzerschnittstellen unterstützt. Auch sie deckt die drei Abstraktionsebenen „Modell“, „AUI“ und „CUI“ ab. Die Einführung von XIML erfolgte im Jahr 1999 und es konnten keine Anzeichen späterer Aktualisierungen aufgefunden werden.

XUL hinterließ den Eindruck der einfachen und effektiven Anwendbarkeit. Es deckt die Abstraktionsebene „CUI“ ab und ist auf jeglicher Plattform, auf der die *Rendering Engine Gecko* läuft, verfügbar. XUL wurde im Jahr 1998 eingeführt und die letzte bisherige Aktualisierung fand 2012 statt.

Die vollständige Beschreibung der Untersuchungsergebnisse befindet sich in Anhang A.

2.5. Modellbasierte UI-Entwicklungsumgebungen

Im Rahmen der Analysen zur vorliegenden Dissertation wurden in einem Literatur-Review 18 modellbasierte Entwicklungsumgebungen für Benutzeroberflächen (Model-Based User Interface Development Environments, MB-UIDE) untersucht. In Unterkapitel 2.5.1 sind folgende MB-UIDE zusammenfassend beschrieben: *User Interface Design Environment* (UIDE), *Interactive Transaction System* (ITS), *High-level UIMS for Manufacturing Applications Needing Organized Iterative Development* (HUMANOID), *Advanced Design Environment for Prototyping with Tasks* (ADEPT), *Generator for User Interfaces Using Software Ergonomic Rules* (GENIUS), *Tools for an Interactive Development Environment* (TRIDENT), *Application Modeling Environment* (AME), *Models Allowing Shared Tools and Explicit Representations to Make Interfaces Natural to Develop* (MASTERMIND), JANUS, *Formal User Interface Specification Environment* (FUSE), MECANO, *Task-based Development of User Interface Software* (TADEUS), *Model-Based Interface Designer* (MOBI-D), TEALLACH, *Task Analysis /*

[87] MARIAE Version 1.5.6 [76]

Design / End User Systems (TADEUS), *Transformation Environment for Interactive Systems Representations* (TERESA), SUPPLE/SUPPLE++ und *MARIA Authoring Environment* (MARIAE).

Abbildung 6 gibt eine Übersicht über die zeitliche Einordnung dieser MB-UIDE. Die Pfeile zeigen hierbei inhaltliche Abhängigkeiten an. Der schraffierte Balken repräsentiert den Entwicklungszeitraum des *CAMELEON Reference Framework* (CRF).

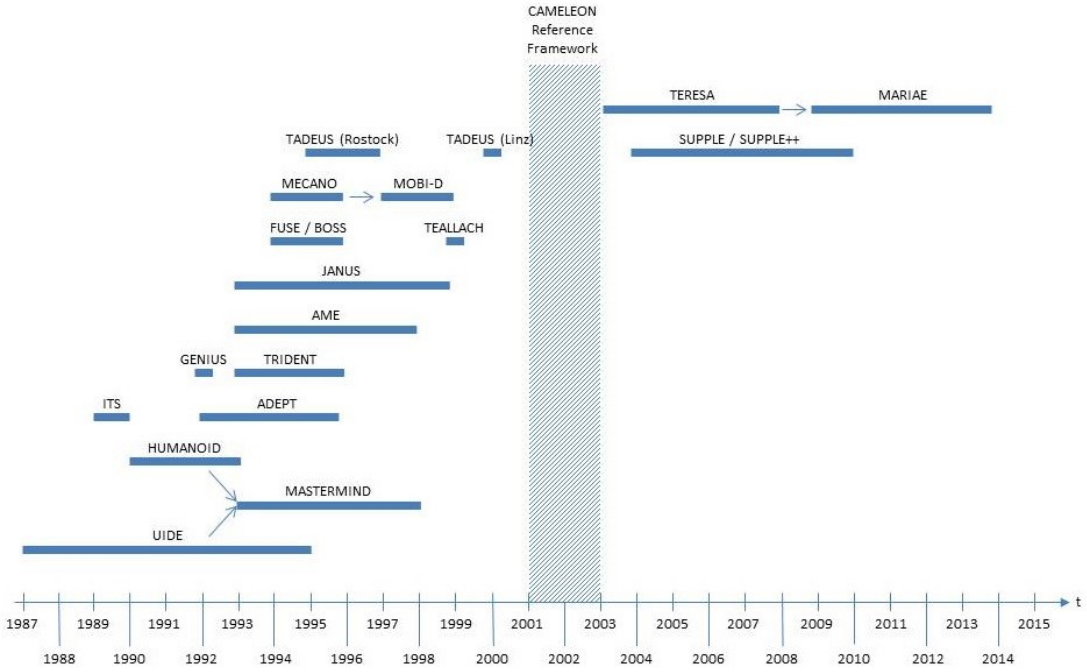


Abbildung 11 ZEITLICHE EINORDNUNG DER UNTERSUCHTEN MB-UIDE

Anschließend erfolgen in Unterkapitel 2.5.2 ein Vergleich und die Bewertung der Sprachen anhand eines zuvor festgelegten Schemas. Insbesondere wurden die jeweils zur Verfügung gestellten Funktionalitäten, die zum Einsatz kommenden Modelle und die Abdeckung der Abstraktionsgrade des CRF (siehe Kapitel 2.1.3) untersucht. Die betreffenden Inhalte wurden bereits teilweise in [78] veröffentlicht.

2.5.1. Untersuchte UI-Entwicklungsumgebungen

2.5.1.1. UIDE

Das *User Interface Design Environment* (UIDE) wurde ab dem Jahr 1987 an der George Washington University in Washington, DC (USA) entworfen und implementiert [79] [80]. Die Entwicklung erfolgte in mehreren Generationen, die sich durch jeweils erweiterte und verbesserte Modellierungskonzepte unterscheiden. Basierend auf den jeweiligen Ausbaustufen der Modelle werden verschiedene zur Entwurfs- oder Laufzeit nutzbare Funktionalitäten, sogenannte Services, angeboten. Dazu zählen Konsistenz- und Vollständigkeitsüberprüfungen, Modelltransformationen, die automatische Erzeugung von Windows, Menüs und Control-Panels, die Aktivierung oder Deaktivierung von Menüeinträgen und anderen Kontrollelementen oder auch die Generierung von textueller oder animierter, kontextabhängiger Hilfefunktionalität [80].

Mit UIDE erfolgt eine wissensbasierte Modellierung, d.h. es wird Metawissen über die zugrundeliegende Softwareanwendung und die zugehörige Benutzerschnittstelle in die Modelle integriert [79] [81] [82] [83]. Zu Beginn unterstützt UIDE ein Applikationsmodell mit zwei Teilmodellen. Einerseits ist dies ein Daten-Modell (*Data Model*), das Informationen über die Datenobjekte der Anwendungsdomäne und deren Eigenschaften beinhaltet und andererseits ein Kontroll-Modell (*Control Model*), in dem die auf diesen Datenobjekten ausführbaren Aktionen beschrieben werden. Darüber hinaus können die Aktionen mit Vor- und Nachbedingungen verknüpft werden [79] [80] [83]. Für jede Aktion werden sogenannte *Parameter* definiert, die die für die Ausführung der Aktion notwendigen Eingabewerte repräsentieren [82].

In späteren Versionen findet dann die Modellierung der Aspekte der zugrundeliegenden Anwendung und der Benutzerschnittstelle in separaten Modellen statt, d.h. es wird zwischen Applikations- und *Interface*-Modell unterschieden [80] [82]. Beide Modelle verfügen jeweils über die bereits genannten Teilmodelle, also Daten- und Kontroll-Modelle. Letzteres wird aber nun als *Operation Model* bezeichnet. Diese Konstruktion ermöglicht eine Unterscheidung der Datenobjekte und Aktionen auf der Anwendungsebene von denen, die seitens der Benutzeroberfläche zur Verfügung stehen. Im Kontroll-Teilmodell des Interface-Modells werden zusätzlich zu den Aktionen auch die möglichen Interaktionstechniken repräsentiert. Diese legen fest, mit welchen Eingabegeräten und in welcher Art und Weise die Interaktionen mit den Interface-Objekten stattfinden [82]. Insgesamt ergeben sich somit drei logische Abstraktionsebenen. Hierbei beschreibt die oberste Ebene die Semantik der Aktionen in der Anwendung, die mittlere Schicht die Aktionen der verschiedenen Interface-Paradigmen in anwendungsunabhängiger Weise und die unterste Ebene die mit den vorhandenen Eingabegeräten möglichen Interaktionen [84].

Im Rahmen der Modellierung erstellt ein Designer zunächst das Applikations-Modell und definiert hierfür die Aktionen und Datenobjekte der Anwendung. Danach oder parallel dazu erfolgt die Beschreibung der Interface-Objekte. Anschließend werden zunächst die als adäquat angesehenen Interface-Aktionen ausgewählt und dann noch die passenden Interaktionstechniken zugeordnet [82] [84]. Die Beschreibungen der Interface-Aktionen stehen vordefiniert in einer Interface Function Library[88] zur Verfügung [82].

Die Speicherung der Modelle erfolgt objektorientiert, d.h. alle Datenobjekte, Aktionen, Parameter und Vor- und Nachbedingungen werden durch Objekte repräsentiert [82]. Diese werden von der Laufzeitumgebung dazu genutzt, um eine ablauffähige Benutzeroberfläche zu erzeugen [82] [84]. Darüber hinaus werden die in den Modellen enthaltenen Informationen von weiteren Werkzeugen verwendet, um die eingangs erwähnten Services zu realisieren. Hierzu zählt das Werkzeug für den Präsentationsentwurf von Benutzerschnittstellen namens *DON*, das dazu dient, automatisiert Vorschläge für Menü-Strukturen und Dialogbox-Repräsentationen zu generieren [85] [86] [80]. Der regelbasierte Assistent *Auto-DevGuide* unterstützt den Designer bei der Erstellung des Interface-Modells hinsichtlich der Auswahl geeigneter Interface-Objekte [80]. Das Tool CARTOONIST nutzt und verfeinert die in den Modellen enthaltenen Informationen, um zur Laufzeit kontextbezogene, animierte Hilfefunktionen zu generieren und dem Anwender zur Verfügung zu stellen [80] [87]. Ein

[88] Engl.: Bibliothek für Benutzerschnittstellenfunktionen

prototypischer Hilfe-Generator erzeugt textbasierte Hilfe-Dialoge, die dem Anwender erklären, warum ein Interface-Objekt im aktuellen Ausführungskontext deaktiviert ist. Das Werkzeug *UIDE System for Semi-Automated GOMS Evaluation* (USAGE) übersetzt die UIDE-Modelle in ein Modell, das die Durchführung von GOMS[89]-Analysen ermöglicht [80].

Im Kontext von UIDE werden auch Benutzer-Modelle verwendet, um adaptive Benutzeroberflächen zu realisieren [80] [81] [84]. Die Erstellung der User-Modelle erfolgt nicht zur Designzeit im Sinne einer Modellierung von angenommenen Eigenschaften von einzelnen Benutzern oder Benutzergruppen. Vielmehr erfolgt zur Laufzeit eine benutzerbezogene Protokollierung der Verwendung der Benutzerschnittstelle, um daraus die Notwendigkeit für dynamische Änderungen der Benutzeroberfläche herzuleiten. Protokolliert werden hierzu verschiedene Interaktions-Statistiken (z. B. erfolgreiche und nicht erfolgreiche Durchführung bestimmter Interaktionen), die chronologische Historie der durchgeführten Interaktionen und die Verwendung der Hilfefunktionen [81].

2.5.1.2. ITS

Das *Interactive Transaction System*[90] (ITS) wurde im Rahmen eines Forschungsprojekts am *T.J. Watson Research Center* der *International Business Machines Corporation* (IBM) in Yorktown Heights (USA) entwickelt. Die erste Veröffentlichung erfolgte im Jahr 1989 [88]. ITS stellt einen regelbasierten Ansatz zur Definition und Generierung von Anwendungs- und Benutzerschnittstellen-Modellen bereit und verfügt über eine Laufzeitumgebung zu deren Ausführung [88]. Unter anderem wurde damit das Besucherinformationssystem der Weltausstellung *EXPO* 1992 in Sevilla implementiert [89], [90].

Mit ITS wird eine strikte Trennung zwischen Präsentations- und Inhaltsebene einer Anwendung verfolgt [88]. Die Architektur von ITS gliedert sich in vier Ebenen [90]:

1. *Action Layer*
   Implementierung der Funktionen der Anwendungslogik. Aktionen lesen, verarbeiten und schreiben Daten ohne die Belange der Dialogsteuerung zu berücksichtigen.
2. *Dialog Layer*
   Definition des Inhalts der Benutzerschnittstelle unabhängig von dessen Repräsentation. Dialoge spezifizieren sogenannte logische *Frames*[91], die in diesen enthaltenen Objekte, die den Objekten zugeordneten Aktionen und den Kontrollfluss zwischen den *Frames*.
3. *Style Rule Layer*
   Definition der Präsentation und des Verhaltens von Familien von Interaktionstechniken. Auf Basis zuvor definierter und jederzeit änderbarer Regeln wird beispielsweise festgelegt, durch welche konkreten Interaktionsobjekte ein abstraktes Interaktionsobjekt repräsentiert werden soll. Die *Style Rule Layer* erstellt für jedes Objekt eines Dialogs ein sogenanntes *Style Program*.

---

[89] GOMS ist eine Analyse-Methode für *Task Performance* und steht für die Begriffe *Goals*, *Operators*, *Methods*, *Selection Rules*, siehe z. B. http://web.eecs.umich.edu/~kieras/docs/TA_Modeling/GOMSforTA.pdf.

[90] Engl.: Interaktives Transaktionssystem

[91] Engl.: Rahmen

4. *Style Program Layer*
   Implementation der Toolkit-Primitive gemäß den Festlegungen in der *Style Rule Layer* und der *Style*-Programme. Während die *Style Rules* zur *Compile*-Zeit ausgeführt werden, kontrolliert die *Style Program Layer* Änderungen in den Dialogen zur Laufzeit. Über das letztendliche Layout wird erst zu dem Zeitpunkt entschieden, zu dem ein *Frame* tatsächlich auf dem Bildschirm dargestellt wird. Das Layout kann sich zu einem späteren Zeitpunkt durchaus noch ändern, wenn beispielsweise der Benutzer die Größe eines Fensters verändert.

ITS sieht einen speziellen Anwendungsentwicklungsprozess vor, für den vier Rollen definiert sind: Anwendungs-Experte, Anwendungs-Programmierer, Style-Experte und Style-Programmierer [88], [90].
Der Anwendungs-Experte definiert den Inhalt der Applikation unabhängig von dessen Präsentation. Dies umfasst sowohl die Definition der Datentypen, die zwischen Benutzerschnittstelle und Anwendung ausgetauscht werden, als auch die Spezifikation der Dialoge [88]. Er benötigt Fachwissen bezüglich der Anwendungsdomäne und muss aber weder Computerfachmann sein, noch Programmierkenntnisse haben [90].
Der Anwendungsprogrammierer entwickelt die Aktionen, die durch die Dialoge aufgerufen werden und die vom Anwendungs-Experten vorgegebenen Datendefinitionen verwenden [88]. Die Aktionen werden in der Programmiersprache *C* realisiert [90].
Ein Dialog-Compiler verarbeitet die Dialogbeschreibungen und Datentypdefinitionen zu einem Syntaxbaum, der an den sogenannten Style-Compiler weitergegeben wird. Dieser benötigt zudem die Style Rules, die vom Style-Experten spezifiziert werden und fügt zu jedem Knoten des Baums die betreffenden Bildschirmobjekte und Interaktionstechniken hinzu. Style-Experten sollten am besten über eine grafische oder softwareergonomische Ausbildung verfügen [88].
Zur Laufzeit interpretiert schließlich ein Dialog-Manager den voll ausgeprägten Syntaxbaum, führt die spezifizierten Aktionen aus und ruft die Bildschirmobjekte in Form von sogenannten *Style Implementation Routines* (SIR) auf. Diese werden vom Style-Programmierer definiert [88].

In [90] werden folgende Vorteile der Schicht-Architektur genannt:

1. Die strikte Trennung von Dialogen und Aktionen erlaubt die Wiederverwendung der Aktionen in anderen Anwendungen.
2. Da die Dialoge unabhängig von ihrer Präsentation spezifiziert werden, können sie durch entsprechende *Style Rules* beliebig dargestellt werden.
3. Änderung der Interaktionstechniken erfolgt in vielen Fällen durch Kombination bereits existierender Toolkit-Objekte mithilfe der *Style Rules* und bedingt keine Neuprogrammierung.

2.5.1.3. HUMANOID

HUMANOID wurde an der *University of Southern California* (USC) in Los Angeles (USA) entwickelt und erstmals im Jahr 1990 veröffentlicht. Der Name steht abkürzend für *High-*

level UIMS for Manufacturing Applications Needing Organized Iterative Development[92]. HUMANOID setzt sogenannte *Templates*[93] ein, anhand derer spezifiziert wird, wie komplexe Anwendungsobjekte in kleinere Einheiten zerlegt und diese Teile schließlich durch grafische Interaktionsobjekte der Zielplattform dargestellt werden können [91].

HUMANOID besteht aus einer modellbasierten Entwicklungs- und einer Laufzeitumgebung, mittels derer die Modelle ausgeführt werden können und somit lauffähige Benutzerschnittstellen zur Verfügung stellt. Es ist das erklärte Ziel, Anwendungs-Designer durch Automatisierung bei der Modellierung der verschiedenen UI-Aspekte auf unterschiedlichen Abstraktionsebenen zu unterstützen, diesen aber bei Bedarf auch die Freiheit lässt, auch auf Detailaspekte zuzugreifen und diese zu manipulieren. Durch die gleichzeitige Anzeige des Modells und des daraus resultierenden UI-Entwurfs ist es möglich Design-Alternativen und –Entscheidungen unmittelbar miteinander zu vergleichen [92]. Zudem ist es mit HUMANOID auch möglich, auf Änderungen, die sich erst zur Laufzeit ergeben, zu reagieren und die Benutzerschnittstelle dynamisch anzupassen [93].

Das deklarative Anwendungsmodell legt fünf verschiedene Aspekte der Anwendung bzw. der Benutzerschnittstelle fest [94]:

1. Applikationssemantik: Anwendungsobjekte und -operationen
2. Präsentation: Äußeres Erscheinungsbild der Benutzerschnittstelle
3. Manipulation: Mögliche Benutzeraktionen für Schnittstellenobjekte und deren Reaktion darauf
4. Dialogabfolge: Reihenfolgebedingungen für Benutzereingaben und Ausführung von Kommandos
5. Seiteneffekte: Nach Benutzeraktionen automatisch durchzuführende Aktionen

Die Präsentation der Benutzerschnittstelle wird inkrementell modelliert. Für jeden Bildschirm werden zunächst die Hauptelemente identifiziert und das Design dann nach und nach verfeinert. Alle Schnittstellenobjekte werden durch abstrakte *Presentation Templates* modelliert, durch welche deren Eigenschaften im Detail festgelegt werden. Sie umfassen folgende Informationen [93]:

1. Spezialisierung: Verfeinerung eines bestehenden Templates durch Änderung seiner vordefinierten Merkmale
2. Eingabedaten: Durch das Template darstellbarer Datentyp
3. Widget: Durch das Template erzeugtes grafisches Objekt eines der folgenden Typen:
    a. Grafisches Primitiv: Linie, Icon, Text, etc.
    b. Toolkit-Primitiv: Menü, Schaltfläche, etc.
    c. Layout Management Widget: Spalte, Zeile, Tabelle, Graph, etc.
4. Anwendbarkeitsbedingungen: Eigenschaften, die den Kontext festlegen, in dem die Anwendung des Templates passend ist
5. Teile: Zerlegung komplexer UI-Objekte in einfachere Teile. Anhand der Eingabedaten eines jeden Teilobjekts wird jeweils ein *Default Presentation Template* ausgewählt, das dieses Teilobjekt modelliert.

[92] Engl.: UIMS höchster Ebene zur Erstellung von Anwendungen mit Bedarf an organisierter iterativer Entwicklungsarbeit
[93] Engl.: Schablonen, Vorlagen

6. Verhalten: Verhalten des durch das Template spezifizierte UI-Objekt bei Eingabe von Daten

Die Verwendung von Default-Werten ermöglicht der Laufzeitumgebung von HUMANOID, auch noch nicht vollständig spezifizierte Modelle bzw. die resultierende Benutzerschnittstelle auszuführen [92]. Mehrfach vorkommende UI-Objekte können durch ein Iterations-Konstrukt namens *Part Replication* modelliert werden. Die Änderung der Darstellung von Objekten in Abhängigkeit von den Inhalten ihrer spezifizierten Eigenschaften kann durch zwei verschiedene Bedingungs-Konstrukte erreicht werden. Einerseits wird durch eine sogenannte *Inclusion Condition*[94] festgelegt, ob das betreffenden Objekt dargestellt wird oder nicht. Die Bedingung wird durch eine Formel realisiert und das Element nur dann dargestellt, wenn die Formel den Wert „wahr" ergibt. Andererseits bieten die *Substitutions*[95] als Liste von Paaren aus Bedingungen und *Presentation Templates* die Möglichkeit, die betreffenden Daten auf alternative Weisen darzustellen [93].

2.5.1.4. ADEPT

Das *Advanced Design Environment for Prototyping with Tasks*[96] (ADEPT) wurde am *Queen Mary and Westfield College* der *University of London* (England) entwickelt. Die erste Veröffentlichung stammt aus dem Jahr 1992. ADEPT ist ein Framework für den aufgabenbasierten Entwurf von Benutzerschnittstellen, die als hierarchische Kompositionen von UI-Objekten angesehen werden. Ausgehend von einem Aufgaben- und einem Benutzer-Modell werden sogenannte *Interface*-Modelle abgeleitet. Dabei handelt es sich zunächst um ein abstraktes Modell der Benutzerschnittstelle, aus dem durch einen Verfeinerungsprozess ein konkretes UI-Modell entsteht. Es bestehen enge Beziehungen zwischen Elementen der Interface- und Aufgaben-Modelle, wenngleich darauf hingewiesen wird, dass die Schnittstellenbeschreibungen UI-Elemente beinhalten können, die keine Pendants im Aufgaben-Modell haben, sondern vielmehr aufgrund von Informationen aus dem Benutzer-Modell ins Design einfließen [95]. Ursprünglich wurde offensichtlich geplant, ADEPT mit einer Laufzeitumgebung zur Ausführung der Benutzerschnittstelle auszustatten [95], in einer späteren Publikation ist aber von plattformabhängigen Implementierungen mit Standard-Widget-Sets wie beispielsweise *Open Look*[97] die Rede [96].

Das Framework beinhaltet verschiedene Werkzeuge wie Editoren, Browser, Interpreter und Generatoren für alle unterstützten Modelltypen. Für die Erstellung der Task-Modelle, die mithilfe von *Task Knowledge Structures*[98] (TKS) beschrieben werden, steht ein graphischer Editor zur Verfügung [96]. ADEPT unterstützt einen komplexen Prozess zur Aufgaben-Modellierung, bei der man schrittweise von einer Task-Analyse über die Definition eines *Existing Task Model*[99] das *Envisioned Task Model*[100] erhält [97]. Die Aufgaben-Modelle werden anschließend von der sogenannten *Abstract Interface Model* (AIM)-Komponente

---

[94] Engl.: Einschluss-Bedingung
[95] Engl: Ersetzungen
[96] Engl: Hochentwickelte Entwurfsumgebung für aufgabenbasierte Prototypen
[97] *Open Look* ist ein eingetragenes Handelszeichen der Firma AT&T Corp. und eine GUI-Spezifikation für Unix-Systeme
[98] Details z. B. in [219]
[99] Engl.: Bestehendes Aufgaben-Modell
[100] Engl.: Angestrebtes Aufgaben-Modell

verarbeitet. Das AIM besteht aus abstrakten Interaktionsobjekten und einer Beschreibung der Dialogstruktur [96], die auf Basis von *Communicating Sequential Processes*[101] (CSP) definiert wird [95]. Auch für das AIM steht ein Editor zur Verfügung. Durch einen weiteren Generator wird aus dem AIM das *Concrete Interface Model* (CIM) erstellt. Das CIM beschreibt die Benutzerschnittstelle auf einer detaillierteren, aber immer noch plattformunabhängigen Ebene und umfasst Informationen sowohl über konkrete Interaktionsobjekte und deren Verhalten sowie das UI-Layout. Auch hier ist für die weitere Bearbeitung ein CIM-Editor vorhanden. Die Generierung des CIM wird maßgeblich durch das Benutzer-Modell beeinflusst [96]. Durch regelbasierte Anwendung von *Design Guidelines* fließen hierbei die Spezifika von Benutzern oder Benutzergruppen in den Entwurf der Benutzerschnittstelle ein [97]. Schließlich wird durch Transformation des CIM das endgültige User Interface auf Basis von *Open Look* erzeugt. Die Unterstützung weiterer Plattformen kann durch die Entwicklung zusätzlicher Generatoren erreicht werden [96].

2.5.1.5. GENIUS

Der *Generator for User Interfaces Using Software Ergonomic Rules*[102] (GENIUS) wurde am *Fraunhofer-Institut für Arbeitswirtschaft und Organisation* in Stuttgart (Deutschland) entwickelt und im Jahr 1993 veröffentlicht. Ausgehend von einem existierenden Datenmodell werden sogenannte Views[103] definiert. Aus diesen wird unter Zuhilfenahme von im System hinterlegten expliziten Design-Regeln das statische Layout der Benutzerschnittstelle erzeugt. Das dynamische Verhalten wird durch Dialog-Netze, die auf der Technik von Petri-Netzen beruhen, modelliert [98].

Die Views bestehen jeweils aus Untermengen von Entitäten, Beziehungen und Attributen des Gesamtdatenmodells, die zur Bearbeitung bestimmter Aufgaben benötigt werden. Sie stellen eine logische Beschreibung der Anwendungsfenster dar. Views können durchaus wiederum andere Views beinhalten. Zudem können zwei verschiedene Arten von Funktionen mit Views verbunden werden. Einerseits sind dies Funktionen zur Datenmanipulation und andererseits Navigationsfunktionen, durch die andere Views aufgerufen werden können und die somit die Dialogstruktur festlegen. Während des Generierungsprozesses werden die Funktionen und deren Eigenschaften dazu verwendet, geeignete Bedienungselemente zu erzeugen. Da nicht alle zur Generierung der Benutzerschnittstelle benötigten Informationen in einer grafischen Repräsentation untergebracht werden können, kommen zusätzliche, textuelle Beschreibungen, die sogenannten *Property Sheets*, zum Einsatz [98].

Die Generierung wird durch eine Wissensbasis unterstützt. Sie enthält zum einen abstrakte Interaktionsobjekttypen, die unabhängig von der späteren physikalischen Implementierung sind. Zum anderen umfasst sie Designregeln zur Auswahl der geeigneten Interaktionsobjekte und Festlegung deren Anordnung. Die eigentliche automatische Erzeugung der Benutzerschnittstelle erfolgt in drei Schritten [98]:

---

[101] Details z. B. in [220]
[102] Engl.: Generator für Benutzerschnittstellen unter Verwendung softwareergonomischer Regeln
[103] Engl.: Sichten

1. Auf Basis der in den Views und Property Sheets enthaltenen Informationen werden die zu verwendenden Interaktionsobjekte ausgewählt und pro View ein Anwendungsfenster generiert.
2. Die Attributwerte für die ausgewählten Interaktionsobjekte werden bestimmt. Die benötigten Informationen hierfür befinden sich in den Property Sheets oder den Beschreibungen der abstrakten Interaktionsobjekttypen.
3. Das Layout wird unter Berücksichtigung der Gesamtheit der Interaktionsobjekte der View bestimmt. Es wird dabei mit der Gestaltung komplexer, zusammengesetzter Elemente begonnen. Die weiteren Elemente werden anschließend gemäß ihrer definierten Reihenfolge oder Wichtigkeit auf der verbleibenden Fensterfläche platziert.

Der Generator erzeugt eine UI-Spezifikation, die von dem zugrundeliegenden User Interface Management System (UIMS) interpretiert werden kann [98].

Um ablauffähige Prototypen der Benutzerschnittstellen zu erhalten, werden zusätzlich zu den erzeugten statischen Fenstern noch Dialogkontrollinformationen benötigt. Hierbei werden zwei Arten der Dialogkontrolle unterschieden. Die sogenannte grobkörnige Dialogkontrolle steuert die Abfolge der Fenster und den Aufruf von aus Benutzereingaben resultierenden Anwendungsfunktionen. Dies ist für erste horizontale UI-Prototypen ausreichend. Sollen vertikale Prototypen, also voll funktionsfähige Benutzerschnittstellen erzeugt werden, so muss eine feinkörnige Dialogkontrolle definiert werden, welche die Zustandsänderungen der einzelnen UI-Objekte beschreibt. Die Dialogkontrolle von GENIUS erfolgt mithilfe der Dialognetze. Diese bestehen wie alle Petri-Netze aus Stellen, Transitionen und Flussrelationen. Views werden den Stellen des Netzes zugeordnet, sodass durch die Markierung von Stellen die dynamische Sichtbarkeit der betreffenden Fenster modelliert wird. Um die Dialognetze überschaubar zu halten, können sogenannte komplexe Stellen eingeführt werden, die eine hierarchische Gliederung in Unterdialoge ermöglichen. Die Anwendungsfunktionen werden als Transitionen im Dialognetz repräsentiert. Der Generator erzeugt aus den Dialognetzen Regeln für das zugrundliegende UIMS [98].

2.5.1.6. TRIDENT

Das Framework namens *Tools for an Interactive Development Environment*[104] (TRIDENT) wurde an den *Facultés Universitaires*[105] *Notre-Dame de la Paix* in Namur (Belgien) entwickelt. Die ersten Veröffentlichungen erfolgten im Jahr 1993 [23] [24]. Das Ziel von TRIDENT ist die automatische Generierung von Benutzerschnittstellen für hochgradig interaktive Geschäftsanwendungen [23] [99], d.h. Anwendungen, durch die große, in Datenbank Management Systemen (DBMS) verwaltete Datenmengen abgefragt und manipuliert werden [24].

Kernstück von TRIDENT ist eine durchgehende Entwicklungsmethodik, die von der Aufgabenanalyse (*Task Analysis*) bis hin zur Generierung einer prototypischen Benutzeroberfläche reicht. Ihre Definition erfolgte unter Berücksichtigung folgender Anforderungen [100]:

[104] Engl.: Werkzeuge für eine Interaktive Entwicklungsumgebung
[105] Franz.: Universitäre Fakultäten

1. Erweiterung bestehender Methoden und Umgebungen zur Anwendungsentwicklung um Aspekte der User Interface-Entwicklung
2. Trennung von Domänen-Funktionalität und Benutzerschnittstelle
3. Trennung von Konversation- und Präsentationskomponenten der Benutzerdialoge
4. Explizite Nutzung ergonomischer Regeln bei der Definition der Präsentation
5. Direkte Ableitung von Konversation und Präsentation aus den Ergebnissen der Task-Analyse

Hierbei bezeichnet die Präsentation (*Presentation*) das statische Layout der Interaktionsobjekte in den Anwenderdialogen und die Konversation (*Conversation*) das dynamische Verhalten der Interaktionsobjekte bei der Manipulation durch den Benutzer [100].

Die resultierende Methodologie besteht aus folgenden Schritten [100]:

1. Durchführung der Aufgabenanalyse
2. Spezifikation der funktionalen Anforderungen
3. Erstellung eines *Activity Chaining Graph*[106] (ACG)
4. Auswahl der Interaktionsart (*Interaction Style*)
5. Festlegung von Präsentationseinheiten (*Presentation Units*, PU)
6. Auswahl von abstrakten Interaktionsobjekten (*Abstract Interaction Objects*, AIO)
7. Transformation der AIO zu konkreten Interaktionsobjekten (*Concrete Interaction Objects*, CIO)
8. Anordnung der konkreten Interaktionsobjekte
9. Generierung der prototypischen Benutzeroberfläche
10. Validierung der Benutzeroberfläche

Im Rahmen der Task-Analyse erfolgt eine Zerlegung der interaktiven Gesamtaufgabe in Teilaufgaben. Bei dieser Dekomposition werden zuvor definierte Benutzer-Stereotypen berücksichtigt. Für diese sind bestimmte Eigenschaften bezüglich Verhalten und Fähigkeiten spezifiziert, wie beispielsweise die Erfahrung mit der Aufgabenstellung (Task Experience), Erfahrung mit dem Umgang mit ähnlichen Anwendungssystemen (System Experience), Grad der Motivation und die Erfahrung hinsichtlich der Verwendung neuer Interaktionstechniken (Media Experience). Diese Merkmale beeinflussen die Ausgestaltung der Teilaufgaben [100].

Die funktionalen Anforderungen beziehen sich auf die Daten- und Funktionsmodellierung. Die Datenmodellierung erfolgt mittels sogenannter *Entity Relationship Attribute* (ERA) Modelle[107]. Die Funktionsmodellierung erfolgt gemäß der Aufgaben-Dekomposition, so dass für jede in der Geschäftslogik der Anwendung vorhandene Aktivität eine Funktion zur Verfügung steht [100].

Auf Basis der Ergebnisse der Task-Analyse und den funktionalen Anforderungen wird ein ACG erstellt, der den Informationsfluss zwischen den Domänen-Funktionen, die zur Erreichung des Aufgabenziels ausgeführt werden müssen, beschreibt. Er spezifiziert für jede vorhandene Funktion die jeweiligen Eingabe- und Ausgabedaten und kennzeichnet, ob diese aus der internen Verarbeitung stammen und vom Benutzer nicht wahrgenommen werden

[106] Engl.: Aktivitätsketten-Graph

[107] Engl.: Gegenstands-Beziehungs-Eigenschafts-Modell. Es beschreibt (Daten-)Objekte, deren Beziehungen untereinander und die Eigenschaften sowohl der Objekte als auch der Beziehungen [215].

können oder aus Systemsicht als extern gelten, also vom Benutzer gesehen und gegebenenfalls manipuliert werden können. Zudem sind logische Verknüpfungen zwischen den Datenobjekten möglich, die in Form von Konjunktion (AND), Disjunktion (OR) und Kontravalenz (XOR) ausgedrückt werden [100].

Die Festlegung des *Interaction Style* erfolgt unter Berücksichtigung der Benutzer-Stereotypen. Gebräuchliche Interaktionsarten sind beispielsweise die natürliche Sprache, Kommando-Sprache, Funktionstasten, Formulare oder direkte Manipulation. Häufig kommen auch Mischformen hiervon zum Einsatz. Die Interaktionsart kann entweder manuell durch einen User Interface-Experten festgelegt oder softwareunterstützt durch Auswertung der Eigenschaften von Aufgaben und Benutzer-Stereotypen automatisiert vorgeschlagen werden [100].

Nach der Fertigstellung des ACG wird die Präsentation im Sinne einer Dialogsteuerung rekursiv in Präsentationseinheiten (PU) zerlegt. Unter anderem wird für jede in der Task-Analyse gefundene Unteraufgabe eine PU definiert. Eine PU setzt sich wiederum aus einem oder mehreren Fenstern (*Windows*) zusammen. Die dynamischen Aspekte werden mithilfe von Zustandsübergangs-Diagrammen (engl. State-Transition Diagrams) beschrieben [100] .

Danach erfolgt die Auswahl von der Situation angemessenen, abstrakten Interaktionsobjekten (AIO) [100]. Angemessen heißt in diesem Zusammenhang, dass sowohl die Gegebenheiten in den Aufgaben- und Datenmodellen als auch die betreffenden Benutzer-Stereotypen und gegebenenfalls auch die Platzverhältnisse des zur Verfügung stehenden Ausgabe-Bildschirms berücksichtigt werden [24]. Die Bestimmung der zu verwendenden AIOs erfolgt mittels Entscheidungstabellen, die aus Gründen einer besseren Übersichtlichkeit und Pflegbarkeit in graphisch repräsentierbare Entscheidungsbäume überführt werden [24] [100]. TRIDENT unterteilt die AIOs in folgende sechs Kategorien: Aktionsobjekte (*Action Objects*, z. B. *Drop-down Menu*), Bildlaufobjekte (*Scrolling Objects*, z .B. *Scroll Bar*), statische Objekte (*Static Objects*, z. B. *Label*), Kontrollobjekte (*Control Objects*, z. B. *Push Button*), Dialogobjekte (*Dialog Objects*, z. B. *Dialog Box*) und Rückmeldungsobjekte (*Feedback Objects*, z. B. *Progression Indicator*) [24] [100] [101].

Anschließend werden die AIOs in konkrete Interaktionsobjekte (CIOs) transformiert. Sie beziehen sich auf die in der betreffenden Zielplattform zur Verfügung stehenden Interaktionselemente. Die Platzierung der CIOs erfolgt entweder statisch automatisiert mittels einer 2-Spalten- bzw. einer Gitterstruktur oder dynamisch computerunterstützt auf Basis eines Koordinatensystems und Heuristiken. Um ungewünschten Effekten vorzubeugen, erfolgt hierbei die Anordnung der Objekte Schritt für Schritt und nachvollziehbar für den UI-Designer. Nach erfolgter Generierung der endgültigen Benutzerschnittstelle kann diese durch Ergonomie-Experten (*Human Factors Experts*) und/oder Endanwender validiert werden [100].

Die Erzeugung der Benutzeroberfläche durch eine iterative Konkretisierung der in den Ausgangsmodellen enthaltenen Informationen über ein abstraktes und ein konkretes Abstraktionsniveau bis schließlich zur finalen Oberfläche nimmt bereits wesentliche Merkmale des später veröffentlichten CAMELEON Reference Framework (siehe Kapitel 2.1.3) vorweg.

Das Architektur-Modell von TRIDENT orientiert sich strikt an der durch das Arch-Modell vorgegebenen Referenz-Architektur (siehe Kapitel 2.1.4). Die domänenspezifische Komponente wird bei TRIDENT durch das eingesetzte DBMS repräsentiert [24].

2.5.1.7. AME

Das *Application Modeling Environment* (AME) wurde an der *Hochschule Augsburg*[108] (Deutschland) entwickelt. Die erste Publikation erfolgte im Jahr 1993 [102]. Die mit AME verfolgte Zielsetzung ist eine enge Integration eines objekt-orientierten Softwareentwicklungs-Prozesses mit der Modellierung und dem Entwurf von Benutzeroberflächen über den gesamten Lebenszyklus des Software-Engineering hinweg. Mit einem Domänen-Modell als Ausgangspunkt nutzt AME dessen strukturelle Informationen, die Beziehungen zwischen den darin enthaltenen Objekten und semantisches Wissen über den Anwendungskontext aus, um prototypische Benutzeroberflächen einschließlich deren dynamisches Verhalten und Anbindung an die Domänen-Objekte automatisch zu generieren [102] [103] [104].

Die Architektur von AME ist in drei Ebenen unterteilt [102] [104]:

1. Modell-Ebene (*Modeling Level*)
2. Konstruktions-Ebene (*Construction Level*)
3. Implementierungs-Ebene (*Implementation Level*)

Auf der Modell-Ebene steht ein CASE[109]-Tool namens *OODevelopTool* zur Verfügung, mit dem das Domänen-Modell in Form eines OOA[110]-Modells erstellt werden kann. Die Aufgabe der Konstruktions-Ebene ist die Verfeinerung des OOA-Modells durch eine Reihe von regelbasierten Werkzeugen, die eine OOD[111]-Struktur erzeugen. Die Implementierungs-Ebene stellt verschiedene Generatoren zur Erzeugung ablauffähiger Benutzeroberflächen zur Verfügung. Einerseits kann C++-Quellcode erzeugt werden, der mittels eines geeigneten Compilers zu übersetzen ist und andererseits gibt es die Möglichkeit, Anwendungen für die Laufzeitumgebung der *KAPPA-PC* Entwicklungsplattform der Firma *Intellicorp Inc.* zu generieren [102] [104].

Unterstützt durch die Werkzeuge auf diesen drei Architekturschichten, besteht der Softwareentwicklungs-Prozess von AME aus folgenden sechs Phasen [103] [104]:

1. Analyse (*Analysis*)
2. Systementwurf (*Global Design*)
3. Detailentwurf (*Detailed Object Design*)
4. Erzeugung der Dynamik (*Generating User Interface Behavior Specification*)
5. Layout und Präsentation (*Adaptation to Specific Users and Environments*)
6. Codegenerierung (*Code Generation and Compilation*)

Bei allen, im Rahmen des Entwicklungsprozesses durch AME automatisch durchgeführten Aktionen besteht generell die Möglichkeit des manuellen Eingriffs durch den Anwendungs-Designer. Dieser erstellt zunächst in der Analyse-Phase das OOA-Modell. Für jede Domänenobjekt-Klasse werden die Attribute, Methoden und Objekt-Beziehungen definiert. AME verwendet für die Repräsentation des Anwendungs-Modells in allen Stadien seines

[108] Damaliger Name *Fachhochschule Augsburg*
[109] Abk.: Computer-Aided Software Engineering
[110] Abk.: Object-Oriented Analysis
[111] Abk.: Object-Oriented Design

Lebenszyklus einen einheitlichen objekt-orientierten Formalismus, der auf den sogenannten *Application System Objects* (ASO) basiert. Das fertige OOA-Modell wird in diese proprietäre Modell-Repräsentation übersetzt. Im Rahmen des Systementwurfs wird das OOA- zu einem OOD-Modell erweitert, das die Fenster-Struktur sowie die Menü- und Befehls-Hierarchien der Anwendung beinhaltet. Dieser Schritt erfolgt durch automatische Generierung, kann aber auch mittels zusätzlicher manueller Eingriffe durch den Designer beeinflusst werden. Während des Detailentwurfs werden allen OOD-Klassen abstrakte Interaktionsobjekte (*Abstract Interaction Objects*, AIO) zugewiesen. Bei der automatischen Auswahl der AIOs werden die Datentypen der Attribute, Kardinalitäten und weitere Meta-Daten ausgewertet. Auch hier ist ein manuelles Ändern und Hinzufügen von Interaktionsobjekten durch den Designer möglich. Die Erzeugung der Dynamik der Benutzeroberfläche erfolgt durch die Auswertung der Eigenschaften der OOD-Objekte und deren adäquate Abbildung auf die ihnen zugeordneten Interaktionsobjekte. In der vorletzten Phase wird die Anpassung an den Benutzer und die Zielplattform vorgenommen. Die Generierung der Präsentation, d.h. die Auswahl geeigneter konkreter Interaktionsobjekte (*Concrete Interaction Objects*, CIO) und die Festlegung derer exakten Ausgestaltung übernimmt ein regelbasiertes Werkzeug. Die Anordnung dieser Objekte wird mithilfe eines Layout-Generators bestimmt. Bei der Codegenerierung wird schließlich der Quellcode in der betreffenden Zielsprache erzeugt [103] [104] [105].

2.5.1.8. MASTERMIND

Die wissensbasierte, integrierte Entwurfs-, Entwicklungs- und Laufzeitumgebung MASTERMIND wurde gemeinschaftlich vom *Graphics, Visualization, and Usability Center* (GVU) des *Georgia Institute of Technology* in Atlanta (USA) und des *Information Sciences Institute* (ISI) der *University of Southern California* in Los Angeles (USA) entwickelt [106] [107] [108] [109]. Der Name MASTERMIND ist ein Akronym und steht für "*Models Allowing Shared Tools and Explicit Representations to Make Interfaces Natural to Develop*"[112]. [106] [80] [110]. Prinzipiell handelt es sich um die Zusammenführung und Weiterentwicklung der beiden komplementären, modellbasierten Ansätze von HUMANOID (siehe Kapitel 2.5.1.3) und UIDE (siehe Kapitel 2.5.1.1) [80] [106] [107]. Die erste Veröffentlichung mit den zugrunde liegenden Ideen stammt aus dem Jahr 1993 [106]. Die eigentliche Implementierung des Frameworks begann Ende 1994 [80]. Erklärtes Ziel von MASTERMIND ist die Definition und Bereitstellung von frei zur Verfügung stehenden Meta-Modellen, die einerseits die bestehenden Tools von HUMANOID und UIDE weiterhin unterstützen und die es andererseits jedermann erlauben, eigene Entwurfs- und Laufzeitwerkzeuge für Benutzeroberflächen zu entwickeln und somit zum MASTERMIND-Framework beizutragen [106].

Die Modelle werden drei verschiedenen Schichten zugeordnet, der semantischen, der syntaktischen und der lexikalischen Ebene. Hierbei sind auf semantischer Ebene Aufgaben- und Applikations-Modelle, auf der syntaktischen Ebene ein Dialog-Modell und auf lexikalischer Ebene Präsentations- und Interaktions-Modelle sowie ein sogenannter *Application Wrapper*

[112] Frei aus dem Englischen übersetzt: Modelle, die durch gemeinsam nutzbare Werkzeuge und explizite Präsentationen die Entwicklung von Benutzeroberflächen auf natürliche Art und Weise ermöglichen

angesiedelt [108] [109]. Im Task-Modell wird für jede Benutzeraufgabe das jeweilige Ziel, die Bedingungen, unter denen die Aufgabe ausgeführt werden kann, der Informationsbedarf, die Auswirkungen der Task-Ausführung sowie eventuelle Unteraufgaben spezifiziert. Das Applikations-Modell definiert die Leistungsmerkmale der zugrunde liegenden Anwendung [107]. Das Dialog-Modell beschreibt die Eingabe-Aktivitäten, die ein Anwender einer durch MASTERMIND generierten Benutzeroberfläche durchzuführen hat sowie deren Auswirkungen auf die Präsentation und die Anwendung. Das Präsentations-Modell spezifiziert alle Konstrukte, die auf dem Bildschirm sichtbar sein können sowie deren Abhängigkeiten untereinander. Das Interaktions-Modell definiert die Menge an Low-Level-Interaktionen zwischen dem Endanwender und der MASTERMIND-Laufzeitumgebung. Diese hängt vom zugrundeliegenden User-Interface-Toolkit ab, für das die Benutzeroberfläche generiert wird. Der *Application Wrapper* ist kein Modell im eigentlichen Sinn, umfasst aber wichtige deklarative Beschreibungen im MASTERMIND-Framework. Durch ihn wird die Schnittstelle zwischen MASTERMIND und der Anwendungslogik festgelegt. Zusätzlich ist noch ein weiteres Modell, das als Kontext-Modell bezeichnet wird, angedacht, das aber bislang nicht realisiert wurde. Es soll durch den UI-Designer spezifizierte Statusinformationen bereit halten, die bei der Ausführung der Benutzeroberfläche berücksichtigt werden müssen [108] [109]. Teilweise werden die Modelle auf Basis von CORBA IDL[113] spezifiziert, so z. B. das Applikations-Modell [107] und der *Application Wrapper* [108] [109]. Das Aufgaben-Modell und dessen Beziehungen zu den anderen MASTERMIND-Modellen werden durch die eigens zu diesem Zweck entwickelte *MASTERMIND Dialogue Language* (MDL) definiert [110].

Der Modellierungsprozess folgt den oben beschriebenen Modell-Schichten. Zunächst werden auf Basis der Ergebnisse einer Task-Analyse das Aufgaben-Modell und dann das Applikations-Modell auf der semantischen Ebene erstellt. Aus den in diesen beiden Modellen enthaltenen Informationen wird anschließend das Dialog-Modell, das sich auf der syntaktischen Ebene befindet, definiert. Danach oder parallel dazu werden die Low-Level-Details in den Präsentations- und Interaktions-Modellen sowie im *Application Wrapper* festgelegt. In diesem Zug werden auch die Verbindungen zwischen den Modellen hergestellt. Abschließend werden durch Generatoren C++-Quellcodedateien erzeugt [108] [109].

2.5.1.9. JANUS

Das JANUS-Framework wurde am Lehrstuhl für Software-Technik der *Ruhr-Universität Bochum* (Deutschland) entwickelt [111] [112] [113]. Die erste Veröffentlichung zu JANUS erfolgte im Jahr 1993 [111]. Ausgehend von einem OOA[114]-Modell (Domänen-Modell) wird durch die Anwendung von Transformationsregeln und unter Berücksichtigung von softwareergonomischen Gesichtspunkten eine grafische Benutzeroberfläche generiert [112] [114]. Hierbei werden C++-Quelldateien erzeugt, die mittels eines geeigneten Compilers übersetzt werden können [114]. Es wird sogar ermöglicht, neben der Anwenderschnittstelle auch noch Teile der zugrundeliegenden Geschäftsanwendung und die Anbindung an eine Daten-

[113] Common Object Request Broker Architecture (CORBA) Interface Definition Language (IDL)
[114] Abk.: Object –Oriented Analysis

haltungsschicht, also an ein Datenbank-Management-System, automatisiert zu erzeugen [113].

Voraussetzung für den Einsatz von JANUS ist, dass im Rahmen der fachlichen Anwendungsmodellierung ein OOA-Modell erstellt wird und dieses vollständig vorliegt. Das Framework beinhaltet hinsichtlich der Modellerstellung keine eigene Unterstützung, etwa in Form eines grafischen Editors. Es können hierfür OO-CASE[115]-Werkzeuge von Drittanbietern genutzt werden. Für einige dieser Werkzeuge wurden Plug-Ins entwickelt, anhand derer die Modelle in das von JANUS benötigte Eingangsformat, in die *JANUS Definition Language* (JDL), übersetzt und als Textdatei zur Verfügung gestellt werden können. Diese Dateien werden von JANUS eingelesen und analysiert [114]. Die so gewonnenen Informationen werden von zwei Generatoren ausgenutzt.

Ein Applikations-Generator erstellt automatisiert Code-Fragmente der Anwendungslogik, die Anbindung an die Datenbank, das Datenbank-Schema und weitere Dienste, wie beispielsweise Druck-Funktionalitäten und ein Hilfe-System [113].

Der GUI[116]-Generator benötigt als Input ein Klassendiagramm, das mindestens die relevanten Klassen mit ihren Attributen und zugehörigen Attributtypen sowie etwaige Vererbungen und Assoziationen umfasst. Für jedes Element des OOA-Modells, also für jede Klasse, jedes Attribut, jede Assoziation usw. kann explizit definiert werden, ob es für die spätere Benutzeroberfläche relevant ist und im Generierungsprozess Berücksichtigung finden soll. Zudem können die Modell-Elemente mit zusätzlichen Informationen versehen werden, so z. B. mit sogenannten ergonomischen Bezeichnern, die später im User-Interface angezeigt werden oder mit Kurzbeschreibungen, die für die kontextsensitive Hilfe verwendet werden können. Grundsätzlich verwendet JANUS bei der GUI-Erzeugung Voreinstellungen, die jedoch durch entsprechende Einträge im Modell überschrieben werden können [114]. Generell wird für jede nicht-abstrakte Klasse ein Fenster erzeugt, das den Namen bzw. den ergonomischen Namen der Klasse trägt. In diesem Fenster werden für jedes Attribut dieser Klasse entsprechende Interaktions-Elemente, die die jeweiligen Attributwerte repräsentieren, erzeugt. Die Label der Interaktions-Elemente, deren Auswahl auf Basis des vorliegenden Attributtyps erfolgt, entsprechen den Namen bzw. den ergonomischen Namen der Attribute. Methoden, die sich auf die Attribute dieser Klasse beziehen, werden in Form von Menü-Einträgen oder als Buttons repräsentiert. Die Platzierung der GUI-Elemente erfolgt mithilfe eines Werkzeugs namens *JANUS Layout Expert System*. Dialogstrukturen werden anhand der bestehenden Beziehungen zwischen Klassen (Assoziation, Aggregation, Einfach- oder Mehrfachvererbung) abgeleitet. Das bei der Generierung der Benutzeroberfläche softwareergonomische Wissen ist in den JANUS Wissensbasen gespeichert. Es werden hierbei Kategorien von Wissen unterschieden [112]:

1. Dialog-Wissen (*Dialog Knowledge*)
   Gestaltungsregeln für verschiedene Typen von Menüs, Menüstrukturen und deren grafische Darstellung

[115] Abk.: Object-Oriented Computer-Aided Software Engineering
[116] Abk.: Graphical User Interface

2. Layout-Wissen (*Layout Knowledge*)
   Gestaltungsregeln hinsichtlich der Struktur von Fenstern, Masken und Interaktionselementen sowie deren Gruppierungen
3. Konventionen (*Conventions*)
   Vordefinierte, anwendungs- oder benutzergruppenbezogene Einstellungen, z. B. die Einhaltung von anwendungsspezifischen Design-Guidelines
4. Selbstbild (Self-descrition)
   Hard- und Software-Konfiguration des Zielsystems, z. B. Bildschirmeigenschaften und zur Verfügung stehende Eingabegeräte

2.5.1.10. FUSE

Das System *Formal User Interface Specification Environment* (FUSE) wurde an der *Technischen Universität München* (Deutschland) entwickelt. Die erste Veröffentlichung erfolgte im Jahr 1996 [115]. Das *Bedienoberflächen-Spezifikationssystem* (BOSS), das später in FUSE integriert wurde, stammt aus dem Jahr 1994 [116]. Im Kontext von FUSE wurde ein Entwicklungsprozess für Benutzerschnittstellen einschließlich eines dazugehörigen Rollen-Modells definiert. Der Prozess besteht aus den drei Phasen Analyse, Design und Evaluation. Für die eigentliche Implementierung der interaktiven Benutzerschnittstelle ist keine eigene Phase vorgesehen, da diese automatisch aus den Entwurfs-Modellen generiert wird [115].

Für jede der Phasen bietet FUSE entsprechende Werkzeugunterstützung an. Die UI-Generierung erfolgt auf Basis von formalen Spezifikationen von Dialog- und Layout-Guidelines. Diese müssen für eine ganze Klasse von Benutzerschnittstellen nur einmal erstellt werden. Deshalb wird hinsichtlich der Architektur von FUSE zwischen den zwei Ebenen *Guideline Definition Layer* (GDL) und *Guideline Application Layer* (GAL) unterschieden. Während in der GDL die Guidelines durch entsprechende Rollen einmalig erstellt werden, finden diese in der Entwurfsphase der GAL jeweils bei der Erstellung bzw. Generierung der Entwurfsmodelle der Benutzerschnittstelle ihre Anwendung. Das FUSE-System umfasst die vier Komponenten *Bedienoberflächen-Spezifikationssystem* (BOSS), *Formal User Interface Development*[117] (FLUID), *Plan-based User Guidance for Intelligent Navigation*[118] (PLUG-IN) und *Formal Interface Requirements Engineering*[119] (FIRE). Mit BOSS wurden verschiedene Bedienoberflächen einer ISDN[120]-Telefon-Simulation für die Firma Siemens AG in Deutschland entwickelt [115].

Im Rahmen der Analysephase werden die formalen Aufgaben-, Problemdomänen- und Benutzer-Modelle mithilfe der Modelleditor-Funktionalität der FIRE-Komponente erstellt. Das Task-Modell ist eine hierarchische Beschreibung der für die Anwendung relevanten Benutzeraufgaben. Das Domänen-Modell ist die algebraische Spezifikation der für das User Interface relevanten Datentypen und Funktionen der Geschäftslogik der interaktiven Applikation. Das Benutzer-Modell beinhaltet Beschreibungen der statischen und dynamischen Eigenschaften einzelner User oder von Benutzergruppen. FIRE erlaubt zudem

[117] Engl.: Formale Benutzerschnittstellen-Entwicklung
[118] Engl.: Planbasierte Benutzerführung für intelligente Navigation
[119] Engl.: Formale Anforderungsanalyse für Schnittstellen
[120] *Integrated Services Digital Network*, Internationaler Standard für Telekommunikationsnetze

die Generierung früher UI-Prototypen, bei denen Elemente der Aufgaben- und Domänen-Modelle in Form von Menüs und Dialogboxen dargestellt werden [115].

In der Entwurfsphase wird zunächst aus den formalen Ausgangsmodellen eine sogenannte logische Benutzerschnittstelle mit bestimmtem Dialog-Stil erzeugt, die die statischen und dynamischen UI-Elemente unabhängig von deren Präsentation beschreibt. Dies geschieht mittels des automatischen Dialog-Designers FLUID unter Anwendung der *Dialog Guidelines*, die die entsprechenden Transformationen definieren. Die Beschreibung der logischen Schnittstelle erfolgt anhand von *Hierarchic Interaction Graph Templates*[121] (HIT), die auf Attributgrammatiken und Fluss-Diagrammen basieren [115]. Detaillierte Informationen zu HITs und deren Anwendung können [116] und [117] entnommen werden.

Anschließend wird durch die BOSS-Komponente unter Zuhilfenahme der *Layout Guidelines*, die ebenfalls mittels der HIT-Technik spezifiziert werden, die Beschreibung der logischen Schnittstelle in eine ausführbare UI-Spezifikation überführt, bei der der Endanwender zur Laufzeit zwischen verschiedenen Layout-Stilen umschalten kann. Darüber hinaus kann das Werkzeug PLUG-IN aus dem Task-Modell und den die Dynamik betreffenden Anteilen der durch FLUID erzeugten logischen Schnittstelle automatisch Inhalte für ein Hilfe- und Benutzerführungssystem generieren. Diese basieren auf *State Transition Diagrams* [122] (STD) und können in die durch BOSS erzeugte Benutzerschnittstelle eingebunden werden [115].

2.5.1.11. MECANO

MECANO wurde an der *Stanford Universität* in Kalifornien (USA) entwickelt. Die erste Veröffentlichung datiert aus dem Jahr 1994 [118]. Bei diesem MB-UIDE wird die grundsätzliche Idee verfolgt, aus einem zum Domänen-Modell erweiterten Datenmodell eine Benutzerschnittstelle automatisch zu generieren. Im Sinne von MECANO ist ein Domänen-Modell eine abstrakte Repräsentation von Objekten in einer Anwendungsdomäne und deren Beziehungen zueinander. Im Vergleich zu einem herkömmlichen Datenmodell ist das Domänen-Modell um weitere Informationen angereichert, sodass auf dessen Basis sowohl das statische Layout als auch das dynamische Verhalten der Benutzeroberfläche generiert werden kann. Die Domänen-Modelle werden, wie auch die aus diesen automatisch generierten Interface-Modelle, in Form von Klassenhierarchien mithilfe einer eigens dafür entwickelten Frame-basierten[123] Beschreibungssprache spezifiziert [118].

MECANO unterstützt den gesamten UI-Lebenszyklus vom Entwurf über die Entwicklung bis hin zur Wartungsphase. Es stehen Modell-Editoren für die beiden Modelltypen, ein Generator zur Erzeugung von lauffähigen Interface-Modell-Instanzen (auch Interface-Spezifikationen genannt) und ein *Interface Builder* zu deren weiteren Bearbeitung zur Verfügung. Zudem existiert ein Laufzeit-System, das zur Ausführung der erzeugten Benutzerschnittstellen benötigt wird. Zur Unterstützung weiterer Laufzeit-Umgebungen für andere Plattformen können textuelle Interface-Spezifikationen generiert werden [118].

---

[121] Engl.: Vorlagen für hierarchische Interaktionsgraphen

[122] Engl.: Zustands-Übergangs-Diagramme

[123] Weitergehende Informationen z. B. unter http://www.obitko.com/tutorials/ontologies-semantic-web/frame-based-models.html

Der iterative Entwicklungsprozess von MECANO besteht aus folgenden fünf Schritten [118]:

1. Erstellung des Domänen-Modells
2. Generierung eines High-Level-Dialogs
3. Generierung des Layouts
4. Generierung eines Low-Level-Dialogs
5. Anpassung des Designs

Nach einer Domänen- und Task-Analyse erstellt der Interface-Entwickler zunächst das Domänen-Modell mittels des Modell-Editors. Aus diesem wird anschließend eine abstrakte Dialog-Spezifikation generiert. Hierzu werden die Informationen in den Klassenhierarchien des Domänen-Modells ausgewertet. Im nächsten Schritt wird das Layout der Benutzerschnittstelle automatisch erzeugt, indem jedem abstrakten Interface-Objekt ein Dialog-Element (entspricht den auf der Zielplattform vorhandenen Interaktionselementen) zugeordnet wird. Die Platzierung der Dialog-Elemente erfolgt unter Berücksichtigung von Design-Guidelines. Die Generierung des Low-Level-Dialogs erfolgt wieder auf Basis der im Domänen-Modell vorhandenen Informationen. Hierbei werden den Dialog-Elementen sogenannte Aktionen (*Actions*) zugewiesen, die das standardmäßige Verhalten der Elemente verändern. Beispiele hierfür sind das Sperren der Editierbarkeit oder die Aktualisierung der Werte anderer Dialog-Elemente nach einer Benutzereingabe. Mit dem nun ablauffähigen Interface führt der Entwickler im letzten Schritt *Participatory Design*-[124]-Sessions mit Endanwendern durch und passt die Benutzerschnittstelle an. Dies geschieht unter Zuhilfenahme des *Interface Builder*-Werkzeugs. Bei größerem Änderungsbedarf kann auch eine Anpassung des Domänen-Modells notwendig werden, sodass ein erneuter Durchlauf durch den oben skizzierten Entwicklungsprozess erforderlich werden kann [118].

Ab 1995 erfolgte eine signifikante Weiterentwicklung von MECANO, die unter dem neuen Namen MOBI-D (siehe Kapitel 2.5.1.12) veröffentlicht wurde [119].

2.5.1.12. TADEUS (Rostock)

Das Framework namens *Task-based Development of User Interface Software* (TADEUS) wurde an der *Universität Rostock* (Deutschland) entwickelt [120] [121]. Die erste Publikation stammt aus dem Jahr 1995 [120]. Die Modellierung der Benutzerschnittstelle erfolgt anhand von Task-, Domänen-, Benutzer- und Dialog-Modellen [120] [121] [22]. Besonderes Augenmerk liegt bei TADEUS auf einer expliziten Dialogmodellierung, die sowohl die statischen als auch die dynamischen Aspekte der Konversation zwischen Mensch und Computer berücksichtigt. Sie erfolgt mittels Dialog-Graphen, die auf Coloured Petri Nets[125] basieren. Das Dialog-Modell übernimmt verschiedene Aufgaben im Entwicklungsprozess. Es stellt die Konsistenz der Modellierung über die verschiedenen Abstraktionsebenen hinweg sicher, wird zur Bewertung von getroffenen Design-Entscheidungen herangezogen und dient als Kommunikationsmittel zwischen Endanwendern und Designer [22]. Eine rudimentäre Version des Dialog-Modells wird mithilfe der in den anderen Modellen enthaltenen

[124] Weitergehende Informationen z. B. unter http://www.maketools.com/articles-papers/FromUsercenteredtoParticipatory_Sanders_%2002.pdf

[125] Weitere Information z. B. unter http://www.informatik.uni-hamburg.de/TGI/PetriNets/index.html

Informationen automatisch generiert und dann Schritt für Schritt verfeinert [120]. Hierbei wird in allen Phasen der Modellierung stets dieselbe Notation verwendet [22].

Grundlage des Dialog-Modells sind sogenannte *Views*[126], die durch den Designer mittels des Aufgaben- und gegebenenfalls auch Domänen-Modells identifiziert und definiert werden. Eine View umfasst zusammengehörende Benutzer-Aufgaben oder -Teilaufgaben aus dem Task- und Objekte aus dem Domänen-Modell [121], die gleichzeitig dargestellt werden sollen [120].

Grundsätzlich unterscheidet TADEUS zwischen Navigations- und Verarbeitungs-Dialogen. Navigations-Dialoge beschreiben hierbei die Übergänge zwischen den Views und werden durch Dialog-Graphen repräsentiert [121]. Verarbeitungsdialoge stellen die Abläufe innerhalb der einzelnen Views dar und werden mittels Interaktionstabellen spezifiziert [120] [121].

Ein Dialog-Graph besteht aus *Nodes*[127], Transitionen und *Token*[128]. Dabei repräsentieren die Nodes die einzelnen Views und die Transitionen die möglichen Interaktionen zwischen diesen. Hinsichtlich der Nodes wird zwischen modalen, non-modalen und komplexen Knoten unterschieden. Die Token beschreiben die Zustände der Views bzw. der aus diesen resultierenden Fenstern der Benutzerschnittstelle. Mögliche Zustände sind „aktiv“ (das Fenster ist sichtbar und wurde aktiviert, d.h. es besitzt den Fokus), „sichtbar“ und „manipulierbar“ (das Fenster ist sichtbar und kann aktiviert werden) [120].

Im Kontext von TADEUS umfasst die Dialogmodellierung sowohl die syntaktische, als auch die lexikalische und semantische Ebene. Die syntaktische Ebene beschreibt hierbei die Struktur und die Reihenfolge der Methoden und Dienste, die zur Zielerreichung benötigt werden. Die lexikalische Modellierung beschäftigt sich mit der Auswahl geeigneter abstrakter und schließlich auch konkreter Interaktionsobjekte, durch die die jeweiligen Domänen-Objekte in der Benutzerschnittstelle repräsentiert werden. Die semantische Modellierung stellt die Verbindung zur eigentlichen Geschäftsanwendung her, indem jede Transition des Dialog-Graphen mit einer Funktion des Applikationskerns verknüpft wird [22].

Anhand des User-Modells werden Eigenschaften von Benutzergruppen, bei TADEUS als Rollen bezeichnet, oder auch einzelnen Benutzern definiert. Es wird zwischen anwendungsunabhängigen und anwendungsabhängigen Merkmalen unterschieden. applikationsunabhängig sind beispielsweise die Fähigkeiten, die kognitiven, visuellen und motorischen Fertigkeiten oder auch Vorlieben der User. Als anwendungsabhängig gelten hingegen zum Beispiel die Benutzer-Ziele und das Wissen über das System und die Anwendung. Diese Informationen werden dazu verwendet, um die Benutzerschnittstelle zur Design-Zeit zu individualisieren [122].

Der UI-Entwicklungsprozess von TADEUS besteht aus folgenden drei Schritten [21]:

1. Anforderungs-Analyse (Requirements Analysis)
2. Dialog-Entwurf (Dialog Design)
3. Automatische Generierung (Automatic Generation)

[126] Engl.: Sichten
[127] Engl.: Knoten
[128] Engl: Marken

Im Rahmen der Anforderungs-Analyse werden die Aufgaben-, Domänen- und Benutzer-Modelle durch den UI-Designer erstellt. Danach wird eine initiale Version des Dialog-Modells generiert und anschließend durch den Designer verfeinert. Im letzten Schritt erfolgt die automatisierte Erzeugung der finalen Benutzeroberfläche mithilfe einer Wissensbasis für Software-Ergonomie. Die Generierung erfolgt dabei in einem siebenstufigen Prozess [21]:

1. Definition und Bewertung der Standard-Layout-Vorgabe
2. Auswahl geeigneter abstrakter Interaktionsobjekte (AIO)
3. Ersetzen der abstrakten durch konkrete Interaktionsobjekte (CIO)
4. Definition des Layouts für die CIO mittels Standardvorgaben
5. Platzierung der CIO
6. Erzeugung des dynamischen Verhaltens
7. Generierung der UI-Beschreibungsdatei für die Zielplattform

Der erste Schritt wird hierbei in der Regel nur einmal und mit Interaktion durch den UI-Designer ausgeführt. Die Schritte 2 bis 7 werden für jede View wiederholt. Die Auswahl von abstrakten und konkreten Interaktionsobjekten erfolgt mittels der Interaktionstabellen. Das dynamische Verhalten wird gemäß den Transitionen im Dialog-Graphen erzeugt. Für die Generierung benötigte Informationen, die nicht in den Ausgangsmodellen vorliegen, werden über einen interaktiven Dialog abgefragt und vom Designer eingegeben. Das Ergebnis ist schließlich eine Dialog-Beschreibung in Form einer Datei, die von einer bestehenden, nicht im TADEUS-Framework enthaltenen, Laufzeitumgebung ausgeführt werden kann [21].

2.5.1.13. MOBI-D

Der *Model-Based Interface Designer* (MOBI-D) wurde ebenso wie die Vorgängerversion namens MECANO (siehe Kapitel 3.4.10) an der *Stanford University* in Kalifornien (USA) entwickelt. Dies erfolgte ab dem Jahr 1995 im Rahmen des Mecano-Projekts, das im Wesentlichen aus zwei Phasen bestand. Zunächst wurde ein generisches Interface-Modell mit dem Ziel definiert, einen hohen Grad an Vollständigkeit zu erreichen, Portabilität zu unterstützen und unabhängig von dem für die Interface-Entwicklung eingesetzten MB-UIDE zu sein. In der zweiten Phase wurde dann ein, dieses Modell unterstützende, MB-UIDE namens MOBI-D entworfen und entwickelt. Ziel war hier eine offene Architektur, um die Entwicklung weiterer Werkzeuge durch Dritte zu ermöglichen [119].

Um die Vollständigkeit hinsichtlich der Beschreibung der zu entwickelnden Benutzerschnittstelle zu erreichen, wurde gegenüber MECANO die Zahl der verwendeten Modelle deutlich erweitert und umfasst nun User-Task-, Domänen-, Präsentations-, Dialog-, Design und User-Modelle [119] [123] [124] [125]. Diese Modelle werden als Teilmodelle des Interface-Modells angesehen und als Modellkomponenten bezeichnet. Hierbei stellt die User-Task-Modellkomponente eine hierarchische Anordnung der ausführbaren Benutzeraktionen dar. Das Domänen-Modell umfasst, wie bei MECANO, alle Domänen-Objekte und deren Beziehungen zueinander. In der Präsentations-Modellkomponente sind alle Interaktions-Elemente enthalten, unabhängig davon, ob sie abstrakter oder konkreter Natur sind. Deren Erscheinungsbild und Lage zueinander wird mittels Attributen und Relationen festgelegt. Das Dialog-Modell ist eine hierarchische Anordnung von durch den Benutzer auslösbaren Kommandos, die den prozeduralen Anteil der Benutzerschnittstelle beschreiben. Das User-Modell enthält die Beschreibungen der Eigenschaften von einzelnen Benutzern oder Benutzergruppen [119] [125]. Nach der Philosophie von MOBI-D lassen sich Design-

Entscheidungen bezüglich der Benutzerschnittstelle als Mappings zwischen Elementen der genannten Modellkomponenten interpretieren [124] [125]. Diese Beziehungen werden, gegebenenfalls verknüpft mit Bedingungen, in der Design-Modellkomponente beschrieben [119].

Das Interface-Modell bzw. dessen oben genannten Modellkomponenten werden mittels einer, an die Konstrukte der Programmiersprache C++ angelehnte, Metasprache namens MIMIC spezifiziert [119] [123]. Prinzipiell müssen Entwickler diese Sprache nicht zwingend im Detail beherrschen, da die Modelle mithilfe der in MOBI-D enthaltenen, visuellen Werkzeuge erstellt und geändert werden können [119]. Zudem ist es nicht nötig, alle Elemente einer Benutzerschnittstelle vollständig neu zu entwickeln, da mit MOBI-D generische Interface-Modelle (bezeichnet als *Mecano Interface Models*, MIM) erstellt werden können, die bereits viele Elemente eines bestimmten UI-Typs enthalten und wiederverwendet werden können [123].

Der Entwicklungsprozess besteht aus folgenden Schritten [124]:

1. Erhebung der Benutzeraufgaben (*User-Task Elicitation*)
2. Erstellung des Task-Modells (*User-Task Modeling*)
3. Erstellung des Domänen-Modells (*Domain Modeling*)
4. Integration der Aufgaben- und Domänen-Modelle (*Task-Domain Integration*)
5. Festlegung von Strategie und Stilart (*Strategy and Style Design*)
6. Erstellung des Präsentations-Modells (*Presentation Design*)
7. Erstellung des Dialog-Modells (*Dialog Design*)
8. Durchführung von Benutzertests (*User Testing*)

Zunächst skizziert ein Domänen-Experte mithilfe des in MOBI-D enthaltenen Werkzeugs *U-TEL* die Benutzeraufgaben in textueller Form. Aus diesen Beschreibungen erstellt und verfeinert der Interface-Entwickler unter Verwendung der Modell-Editoren die User-Task- und Domänen-Modelle. Dies kann parallel zu einander erfolgen. Anschließend spezifiziert er die Beziehungen zwischen den beiden Modellen, d.h. es wird festgelegt, welche Domänen-Objekte für die einzelnen Aufgaben bzw. Teilaufgaben benötigt werden. Optional wird noch ein User-Modell erstellt. Mit diesem Schritt ist die Modellierung auf abstrakter Ebene fertiggestellt und es kann mit der Konkretisierung begonnen werden [124]. Anders als bei MECANO, das eine vollständige Automatisierung hinsichtlich der nun folgenden Design-Entscheidungen zum Ziel hatte, werden bei MOBI-D die Mapping-Möglichkeiten zwar automatisch berechnet und priorisiert, der Entwickler kann diese aber jederzeit beliebig ändern oder verwerfen. In diesem Sinne werden die Benutzerschnittstellen praktisch nicht automatisch generiert, sondern es steht vielmehr ein Design-Assistenz-Werkzeug namens *The Interface Model Mapper* (TIMM) zur Verfügung, das als *Decision Support System*[129] (DSS) fungiert [125]. Es unterstützt den Entwickler bei seiner Arbeit und überlässt ihm deutlich mehr Einflussmöglichkeiten und Gestaltungsfreiheiten [123]. Auf diese Weise werden nun zunächst die Stilarten, Guidelines und Dialog-Strategien bestimmt und die Präsentations- und Dialog-Modelle entworfen. Danach wird dann noch das Design, gegebenenfalls unter Mitarbeit des Domänen-Experten oder Endanwendern, mit einem aufgabenbasierten *Inter-*

[129] Engl.: Entscheidungsunterstützungssystem (EUS)

face Builder finalisiert. Schließlich können die Tests der so entstandenen Benutzeroberfläche durchgeführt werden [124].

2.5.1.14. TEALLACH

Die modellbasierte UI-Entwicklungsumgebung TEALLACH ist eine gemeinsame Entwicklung der *University of Manchester* (UK), der *Napier University* in Edinburgh (UK) und der *University of Glasgow* (UK) und wurde im Jahr 1999 publiziert. Bei diesem Ansatz wird davon ausgegangen, dass die zugrundeliegende Anwendung, zu der die Benutzerschnittstelle entwickelt werden soll, bereits vorhanden ist. Zur UI-Generierung werden drei Modelle benötigt, nämlich ein Domänen-, ein Aufgaben- und ein Präsentations-Modell [126].

Um eine Plattform-Unabhängigkeit zu erreichen, basiert das *Domain Model* auf den im ODMG[130] Objekt-Datenbank-Standard spezifizierten Konzepten. Im Wesentlichen handelt es sich um ein Meta-Modell, dass jedes der ODMG-Konzepte als instanziierbare Klasse realisiert. Ähnlich wurde auch bei allen Konstrukten des ODMG-Objektmodells vorgegangen. Ergebnis ist eine Sammlung von Klassen, die als Bausteine für Domänen-Modelle angesehen werden können. Die hauptsächliche Aufgabe des Domänen-Modells ist die Repräsentation der zugrundeliegenden Anwendung, der Datenbank-Konnektivität und –Interaktion. Zudem können Hilfsdatentypen modelliert werden, die dazu dienen, zur Laufzeit benötigte transiente Daten zu beschreiben [126].

Das Aufgaben-Modell unterstützt sowohl die Modellierung der Struktur der Aufgaben der Benutzer, als auch des Informationsflusses zwischen den Modellen, während der Aufgabenausführung. Es wird repräsentiert durch einen hierarchischen Baum, dessen Blätter sogenannte Aktions- bzw. Interaktionsaufgaben darstellen. *Action Tasks* sind hierbei elementare Aufgaben, die von der Anwendung durchgeführt werden, während bei *Interaction Tasks* entweder der Benutzer Eingaben macht oder der Computer Rückmeldung an den User liefert. Darüber hinaus sind folgende temporale Beziehungen spezifizierbar: sequenziell (in der vorgegebenen Reihenfolge), reihenfolgeunabhängig, wiederholbar (entsprechend einer vorgegebenen Anzahl von Wiederholungen oder bis eine definierte Bedingung erfüllt ist), nebenläufig, Auswahl (durch den Benutzer, der festlegt, welcher der spezifizierten Unteraufgaben durchgeführt werden soll), optional und bedingt (Wahl zwischen Unteraufgaben auf Basis einer gegebenen Bedingung). Ferner erlaubt das Aufgaben-Modell, lokale Zustände zu deklarieren und diese mit einer Aufgabe zu assoziieren. Auf diese Weise können Informationsflüsse modelliert werden [126].

Das Präsentations-Modell beschreibt das Erscheinungsbild und das Verhalten der Benutzerschnittstelle. Es gibt hierbei zwei Abstraktionsebenen: das konkrete und das abstrakte Präsentations-Modell. Die Elemente des konkreten Präsentations-Modells entsprechen den verfügbaren Widgets[131] der Implementierungsumgebung. TEALLACH verwendet hierbei die UI-Elemente von *Java Swing*[132]. Das abstrakte Präsentations-Modell spezifiziert abstrakte Kategorien, denen konkrete GUI-Elemente zugeordnet werden können. Beispielsweise ist

[130] Object Data Management Group, siehe http://www.odbms.org/odmg/
[131] Elemente einer grafischen Benutzeroberfläche
[132] Teil der Java Foundation Classes (JFC) der Firma Oracle Inc., Application Programming Interface (API) zur Entwicklung von grafischen Benutzerschnittstellen für Java-Programme

die abstrakte Kategorie *Chooser*[133] definiert, die auf der konkreten Ebene durch Kontroll-Elemente wie *Radio Buttons*, *Drop-down List* oder *Scrollable List* repräsentiert werden kann [126].

Im Zusammenspiel der drei vorher beschriebenen Modelle können Assoziationen zwischen Elementen der verschiedenen Modelle auf zwei Arten definiert werden: durch *Linking*[134] werden Verbindungen zwischen zwei verschiedenen Modellen erzeugt, während durch *Deriving*[135] Teile des einen Modells auf Basis von Informationen aus dem anderen erzeugt werden. Ein Vorteil von TEALLACH ist, dass bei der Erstellung der Modelle keine strikte Reihenfolge eingehalten werden muss. So können während der Modellierung auf die oben beschriebene Weise Beziehungen zwischen Domänen-, Aufgaben- und Präsentations-Modell spezifiziert werden. Einzige Ausnahme ist das Domänen-Modell, das als nicht änderbar angesehen wird und somit nicht durch *Derive*-Operationen beeinflusst werden darf [126].

2.5.1.15. TADEUS (Linz)

Neben der in Kapitel 2.5.1.12 beschriebenen Entwicklungsumgebung gibt es eine weitere MB-UIDE mit der Bezeichnung TADEUS, die im Rahmen des gleichnamigen Projekts von 1997 bis 2000 an der *Johannes Kepler Universität* in Linz (Österreich) entwickelt wurde [127]. Der Name TADEUS steht hierbei für *Task Analysis / Design / End User Systems* [127] [128]. Dieser Ansatz verfolgt eine werkzeugunterstütze Entwicklung von prototypischen Benutzerschnittstellen bei gleichzeitiger Berücksichtigung sowohl der Systemfunktionalitäten als auch von Usability-Anforderungen [127].

TADEUS unterstützt gleichermaßen Domänen-Experten, Analysten, Designer, Usability-Experten, Programmierer und Endanwender durch einen kombinierten Ansatz aus [128]:

1. Modellgetriebener Entwicklung (Model-driven Development)
2. Aufgabenbasierter Entwicklung (Task-based Development)
3. Benutzerorientierter Entwicklung (User-oriented Development)
4. Objektorientierte Entwicklung (Object-oriented Development)

Hierbei erlaubt die Modellgetriebenheit eine multidimensionale und perspektivenbezogene Spezifikation der Benutzerschnittstelle [128]. Das Referenz-Modell von TADEUS besteht aus Aufgaben-, Benutzer-, Daten-[136] und Interaktions-Modell [127]. Diese vier Teilmodelle werden zusammengenommen auch als Anwendungs-Modell (Application Model) bezeichnet. Das UI-Design wird als kontinuierlicher Wechsel zwischen diesen Perspektiven angesehen. Durch die Aufgabenbezogenheit wird nicht nur die Anwendung selbst, sondern auch deren ganzer organisatorischer und geschäftlicher Kontext mitbetrachtet. Die Benutzerorientierung erlaubt es, individuelle Fähigkeiten und Vorlieben der Endanwender zu berücksichtigen. Die Objektorientiertheit ermöglicht schließlich eine einheitliche und diagrammatische Repräsentation aller relevanten Fakten, die die Kommunikation zwischen den oben genannten Rollen und die Implementierung von prototypischen Benutzeroberflä-

[133] Abgeleitet aus den engl. Wort „to choose“ (wählen). Möglichkeit zur Auswahl einer Option aus eine Menge von mehreren.
[134] Engl:. Verbinden
[135] Engl.: Ableiten
[136] Im Kontext von TADEUS auch als *Problem Domain*-Modell bezeichnet

chen erleichtert. Sie basiert auf Ausdrucksmitteln der objektorientierten Analyse (Object-oriented Analysis, OOA) in Form von [128]:

1. Objekt-Beziehungs-Diagrammen (Object Relationship Diagrams, ORD)
2. Objekt-Verhaltens-Diagramme (Object Behavior Diagrams, ODB)
3. Objekt-Interaktions-Diagrammen (Object Interaction Diagrams, OID)

Mittels ORD lassen sich strukturelle Beziehungen zwischen Klassen oder Objekten beschreiben, während durch ein OBD die dynamischen Eigenschaften von Objekten mithilfe von Zuständen und Zustandsübergängen (Transitionen) definiert werden. Ein OID legt die möglichen Interaktionen zwischen den in den OBDs spezifizierten Objekten fest [128].

Das Vorgehen bei der Entwicklung von Benutzerschnittstellen mit TADEUS lässt sich in folgende drei grundlegende Schritte einteilen [128]:

1. Analyse
2. Design
3. Implementierung

Die Analyse erfolgt unter Zuhilfenahme der *TADEUS Task Analysis and Representation* (TATAR) Technik. Mit dieser Methode werden nicht nur die Organisation und die Geschäftsprozesse inklusiver aller zu deren Durchführung benötigten Aktivitäten analysiert und beschrieben, sondern auch weitergehende Aspekte wie beispielsweise die Arbeitsumgebung, soziale Gegebenheiten, für die Ausführung der Aktivitäten benötigten Fähigkeiten und Wissen, physische und kognitive Fertigkeiten und Erfahrungen. All diese Informationen werden in ein einheitliches Modell, das als *Business Intelligence*-Modell bezeichnet wird, überführt [128].

Auf Basis des *Business Intelligence*-Modells erstellt ein Designer die Task-, User-, Domänen- und Interaktionsmodelle und verfeinert diese schrittweise. Es wird dabei jeweils zwischen struktureller und dynamischer Modellierung unterschieden. Bei der strukturellen Aufgaben-Modellierung werden diejenigen Benutzer-Aktivitäten ausgewählt, die durch die zukünftige Anwendung unterstützt werden sollen. Die dynamische Task-Modellierung legt die temporalen Abhängigkeiten dieser Aktivitäten sowie deren Ein- und Ausgabe-Verhalten fest. Innerhalb der strukturellen User-Modellierung werden sowohl funktionale Gruppierungen von Benutzern als auch deren individuelle Eigenschaften festgelegt. Bei der dynamischen Benutzer-Modellierung wird der Arbeitsablauf aus der Sicht der verschiedenen funktionalen Rollen beschrieben. Die strukturelle Daten-Modellierung befasst sich mit der Definition von Klassen, die für die Ausführung der Aktivitäten benötigt werden, während die dynamische Modellierung hierbei den Lebenszyklus dieser Klassen festlegt. Im Rahmen der strukturellen Interaktions-Modellierung werden Anordnung und Darstellungsstil von generischen oder plattformspezifischen Interaktionselementen spezifiziert. Die Definition deren dynamischen Verhaltens erfolgt schließlich im Kontext der dynamischen Interaktions-Modellierung. Die gesamten Design-Aktivitäten werden durch eine TADEUS-Komponente namens *Interactive Workspace* unterstützt. Zudem ermöglicht ein als *Consistency Checker* bezeichnetes Werkzeug die automatisierte Überprüfung der Konsistenz des Zusammenspiels der Modell-Komponenten [128].

Das Anwendungs-Modell kann durch die sogenannte *Prototyping Engine* interpretiert und als prototypische Benutzerschnittelle ausgeführt werden. Es besteht die Möglichkeit, das Applikations-Modell und somit den Programmablauf zur Laufzeit zu verändern [128].

2.5.1.16. TERESA

Das *Transformation Environment for Interactive Systems Representations* (TERESA) wurde, wie auch TERESA XML (siehe Kapitel 2.4.1.7) von der HCI-Gruppe des ISTI-CNR (Italien) entwickelt. Die erste Veröffentlichung geht auf das Jahr 2003 zurück. Bei dieser ersten TERESA-Version wird zunächst ein sogenanntes *High-level Task Model* erstellt, das neben den eigentlichen Aufgaben auch Informationen über die verschiedenen Benutzungskontexte und die beteiligten Rollen beinhaltet. Zudem umfasst es ein Domänen-Model, das sowohl sämtliche Objekte, die bei der Ausführung der Aufgaben manipuliert werden, als auch deren Beziehungen untereinander, beschreibt. Aus diesem werden anschließend durch Filterung und eventuelle Verfeinerung plattformspezifische *System Task Models* abgeleitet. Auf deren Basis entsteht ein *Abstract User Interface* (AUI), das aus einer Anzahl von abstrakten Präsentationen besteht, die sich durch Analyse der Relationen zwischen den einzelnen Teilaufgaben ergeben. Die Präsentationen wiederum sind Kompositionen von abstrakten Interaktionsobjekten, die mittels der Operatoren „Gruppierung“, „Reihenfolge“, „Hierarchie“ und „Beziehung“ beschrieben werden. Auf Basis des AUI wird dann eine plattformabhängige Benutzerschnittstelle generiert, die alle Spezifika des verwendeten Endgeräts und dessen Betriebssystems berücksichtigt. Hierbei wird jedes abstrakte Interaktionsobjekt durch ein konkretes ersetzt [74]. Aus dieser konkreten Beschreibung wird schließlich die endgültige Benutzerschnittstelle generiert, z. B. in Form von XHTML- oder Java-Code [62].

Aufgrund der Erkenntnis, dass es weder möglich noch gewünscht ist, alle Teilaufgaben des Task-Modells auf allen Zielplattformen in der gleichen Art und Weise zu implementieren, werden bei TERESA folgende Fälle unterschieden [62]:

1. Aufgaben, die auf mehreren Plattformen auf die gleiche Weise realisiert werden
2. Aufgaben, die nur auf einer Plattform sinnvoll umgesetzt werden können
3. Abhängigkeiten zwischen Aufgaben, die auf unterschiedlichen Plattformen durchgeführt werden
4. Aufgaben, die auf mehreren Plattformen zur Verfügung stehen, aber unterschiedlich implementiert sind:
   a. mittels verschiedener Domänen-Objekte
   b. mittels verschiedener Benutzerschnittstellen-Objekte
   c. mittels verschiedener Aufgaben-Dekompositionen
   d. mittels verschiedener temporaler Beziehungen zwischen den Unteraufgaben

Das TERESA-System unterstützt den UI-Designer beim Fällen der diesbezüglichen Entscheidungen durch passende Vorschläge. Die Konsistenz des Schnittstellenentwurfs wird durch die Anwendung der stets gleichen Designkriterien innerhalb der Transformationen zwischen den einzelnen Abstraktionsebenen erreicht [58].

Im Kontext von TERESA wird der Begriff „Plattform“ als eine Menge interaktiver Endgeräte mit ähnlichen Eigenschaften verstanden, also etwa Desktop-Systeme oder Mobiltelefone. Im Zuge einer Weiterentwicklung wurde zur bestehenden Unterstützung mehrerer Plattformen die zusätzliche Erreichung von Multimodalität angestrebt. Das heißt, dass spezifische Interaktionstechniken, wie beispielsweise Gestensteuerung, explizit modelliert werden können. Für folgende Plattformen/Modalitäten wurden mit TERESA bereits Benutzerschnittstellen implementiert: Desktop-Systeme, Systeme mit interaktiver Vektorgrafik, vokale und grafische Multimodalität, digitales TV sowie gestische und grafische Multimodalität. Als

Zielsprachen wurden bislang *XHTML Mobile Profile*[137] (MP), *VoiceXML*[138], *X+V*[139], *SVG*[140], *Xlet*[141] und die Gestenbibliothek der Firma *Microsoft* realisiert [73].

2.5.1.17. SUPPLE und SUPPLE++

Das SUPPLE[142]-System wurde an der *University of Washington* in Seattle (USA) entwickelt. Die erste Veröffentlichung geht auf das Jahr 2004 zurück [129], das aktuellste aufgefundene Dokument stammt aus dem Jahr 2010 [130]. Dieser Ansatz wertet *Interface*-Spezifikationen sowie *Device*- und *User*-Modelle aus. Die Erzeugung und Anpassung von Benutzeroberflächen wird auf Basis von Lösungen zu Optimierungsproblemen behandelt. SUPPLE sucht nach Darstellungen, die den durch das verwendete Endgerät entstehenden Einschränkungen gerecht werden und den erwarteten Aufwand für die Ausführung der Benutzeraktionen minimiert [129]. Zudem passt SUPPLE die Benutzerschnittstelle an den persönlichen Arbeitsstil des Users an [131] und implementiert bei Bedarf individuelle Präferenzen des Benutzers [132]. Die Generierung und Anpassung der Oberflächen erfolgt zur Laufzeit [131]. SUPPLE++ ist eine Variante des SUPPLE-Systems und unterstützt die automatische Erzeugung und Anpassung von User Interfaces für Benutzer mit Sehbehinderungen und/oder eingeschränkten motorischen Fähigkeiten. SUPPLE++ wird erstmals in einer Publikation aus dem Jahr 2007 beschrieben [133].

Eine Interface-Spezifikation ist definiert durch eine Menge von abstrakten Schnittstellen-Elementen und eine Menge von die Schnittstelle betreffenden Einschränkungen, die durch den UI-Designer vorgegeben werden. Die Elemente werden anhand ihres Datentyps spezifiziert. Zulässig sind hierbei primitive Datentypen (*integer*, *float*, *string* und *boolean*, aber auch Konstrukte für Datum, Uhrzeit, Bilder und Karten), einfache Datentypen mit eingeschränktem Wertebereich, Vektoren (geordnete Sequenz von Werten gleichen Typs), Container (Sequenz von Elementen) und sogenannte Aktionstypen zur Ausführung von Methoden [134]. Die schnittstellenspezifischen Einschränkungen werden mittels Funktionen definiert, die ein vollständiges oder teilweises Rendering auf einen Booleschen Wert abbilden. Somit kann beispielsweise sichergestellt werden, dass bestimmte Elemente durch dasselbe Widget dargestellt werden [129].

Das Device-Modell beinhaltet die Menge, der für das betreffende Gerät zur Verfügung stehenden Widgets, eine Anzahl von gerätespezifischen Einschränkungen und zwei geräteabhängige Funktionen zur Bewertung der Eignung der zu verwendenden Interaktionsobjekte. Wie schon zuvor, werden die Einschränkungen, mit denen sich beispielsweise die zur Verfügung stehende Bildschirmgröße modellieren lässt, auch hier als Boolesche Funktionen angegeben. Durch sie wird definiert, ob Widgets im gegebenen Kontext benützt werden können oder nicht. Die geräteabhängigen Funktionen spiegeln den geschätzten Aufwand

---

[137] XHTML angepasst zur Verwendung auf Mobiltelefonen und *Personal Digital Assistants* (PDA)

[138] Voice Extensible Markup Language, Sprache zur Beschreibung von Dialogabläufen in sprachbasierten Systemen

[139] XHTML+Voice Profile, vom *World Wide Web Consortium* (W3C) empfohlene Sprache zur Beschreibung multimodaler Benutzerschnittstellen, siehe http://www.w3.org/TR/xhtml+voice/

[140] Scalable Vector Graphics, vom W3C empfohlene Spezifikation zur Beschreibung zweidimensionaler Vektorgrafiken, siehe http://www.w3.org/TR/SVG11/

[141] Spezifikation zur Unterstützung von Anwendungen für digitales TV

[142] Engl.: biegsam, geschmeidig

wider, den der Benutzer betreiben muss, um die Widgets der Benutzeroberfläche zu bedienen. Die erste Funktion sagt aus, wie geeignet jedes verwendete Interaktionsobjekt für die Repräsentation des betreffenden abstrakten Elements aus der Interface-Spezifikation ist. Die zweite dient der Abschätzung des Navigationsaufwands, der für die Bedienung der Benutzerschnittstelle benötigt wird. Die beiden Funktionen werden in Form einer sogenannten Kostenfunktion in der weiteren Verarbeitung in SUPPLE verwendet [129].

Das User-Modell wird mithilfe sogenannter *User Traces*[143] definiert. Unter einem *User Trace* wird hierbei eine Anzahl von *User Trails*[144] verstanden. Ein *Trail* wiederum ist eine zusammenhängende Sequenz von Elementen, die durch den Benutzer manipuliert wurden. Auf diese Weise wird das Benutzerverhalten im User-Modell repräsentiert. Diese Information kann beispielsweise auf einem Endgerät gesammelt werden und dann die Generierung der Benutzerschnittstelle auf einem anderen Device beeinflussen [129].

Ziel von SUPPLE ist, jedes abstrakte Interaktionselement der Interface-Spezifikation durch ein geeignetes Widget der aktuell verwendeten Geräte-Plattform zu rendern. Hierzu werden durch einen Algorithmus sogenannte erlaubte Renderings ermittelt und daraus das Beste ausgewählt. Das beste Rendering ist jenes, welches die Kostenfunktion, die die beiden vorher beschriebenen Funktionen des Device-Modells beinhaltet, minimiert. Hierzu steht ein optimierter *Branch-and-Bound*-Algorithmus zur Verfügung, der soweit optimiert ist, dass er auch auf leistungsschwächeren Endgeräten eingesetzt werden kann [129]. Grundsätzlich verfügt SUPPLE aber über eine verteilte Architektur, sodass der Render-Service, der hier als *Solver Server* bezeichnet wird, auch auf einem entfernten Gerät laufen kann und durch eine Netzwerkverbindung erreichbar sein muss [134].

Die Kostenfunktionen bestehen typischerweise aus mehr als 40 Parametern, die aufeinander abgestimmt sein müssen. Zunächst erfolgte die Auswahl der Werte für diese Parameter manuell, was jedoch eine langwierige und fehleranfällige Angelegenheit darstellt. Deshalb wurde zur Automatisierung dieses Prozesses ein Werkzeug namens ARNAULD[145] entwickelt [132]. Hierbei werden zwei unterschiedliche Techniken eingesetzt. Einerseits bietet sich beim *Example Critiquing*[146] dem Benutzer die Möglichkeit, auf dem Desktop-Computer mit einem Rechts-Klick auf jedes beliebige Interface-Element ein Dialogfenster zu öffnen, eines der dort angebotenen, möglichen alternativen Renderings auszuwählen und somit die Darstellung des Elements zu verändern. Diese Anpassungswünsche werden von ARNAULD aufgezeichnet. Andererseits werden bei der *Active Elicitation*[147] dem Benutzer im Sinne einer Befragung mehrmals jeweils zwei unterschiedliche Renderings zu einem bestimmten abstrakten Interaktionsobjekt angezeigt und abgefragt welches davon er gegebenenfalls präferiert. Die Abfragen hierbei automatisch generiert und die Ergebnisse aufgezeichnet. Die Erkenntnisse aus den beiden Erhebungsmethoden fließen dann in die Findung der Kostenfunktion ein [131].

SUPPLE++ basiert auf SUPPLE und modelliert die motorischen [135] und visuellen Fähigkeiten von Benutzern mit Behinderungen [133]. Im Wesentlichen mussten zur

[143] Engl.: Spur
[144] Engl.: Pfad
[145] Benannt nach dem französischen Philosophen Antoine Arnauld (1612-1694) [131]
[146] Engl.: Kritische Abhandlung am Beispiel
[147] Engl.: Aktive Erhebung

Modellierung des jeweils individuellen Vermögens der Benutzer andere, noch komplexere Kostenfunktionen für die Optimierungsaufgaben gefunden werden [133]. Aus den gleichen Gründen, die zur Implementierung von ARNAULD führten, wurde für SUPPLE++ der sogenannte *Activity Modeler* entwickelt. Mithilfe dieses Werkzeugs muss ein User vor der Benutzung der Oberfläche einmalig einen Performance-Test durchführen, aus dessen Ergebnissen ein benutzerspezifisches Regressionsmodell hinsichtlich seiner motorischen Fähigkeiten erstellt werden kann. Es sind hierbei verschiedene Testaufgaben hinsichtlich Zeigen (Pointing), Ziehen (Dragging), Listenauswahl und Mehrfach-Klicken mit verschiedenen Interaktionsobjekten in verschiedenen Nutzungskontexten, wie z. B. unterschiedliche Größen der Darstellung der Objekte, zu lösen. Dabei werden die zur Navigation und Manipulation der Widgets benötigten Zeiten, die für den Testdurchgang benötigte Gesamtzeit und die Fehlerraten gemessen [135]. Auf Basis der Ergebnisse wird die Benutzeroberfläche automatisch für den Benutzer individuell angepasst [132]. Im Rahmen von Versuchen konnten signifikante Verbesserungen der Ausführungszeiten und der subjektiven Benutzerzufriedenheit bei mit SUPPLE++ angepassten im Vergleich zu herkömmlichen Schnittstellen festgestellt werden [135].

2.5.1.18. MARIAE

Das *MARIA Authoring Environment* (MARIAE) wurde, wie auch die zugehörende Beschreibungssprache für Benutzeroberflächen *MARIA* (siehe Kapitel 2.4.1.8) von der HCI-Gruppe des *ISTI-CNR* in Pisa (Italien) entwickelt [136]. Die initiale Version wurde auf Basis der mit dem Vorgänger *TERESA* (siehe Kapitel 2.5.1.16) gesammelten Erkenntnisse und Erfahrungen konzipiert [60]. Die ersten Veröffentlichungen gehen auf das Jahr 2009 zurück [64] [65] [137]. MARIAE wird bis heute noch immer weiterentwickelt; die aktuelle Version 1.5.6[148] ist im Internet frei erhältlich.

Folgende hauptsächlichen Erweiterungen und Verbesserungen gegenüber TERESA wurden verfolgt [64]:

1. Einführung eines Event-Modells, um Designern mehr Einfluss auf die letztlich erzeugte Benutzerschnittstelle zu erlauben
2. Mehr Flexibilität bezüglich des Dialog-Modells, das nun auch parallele Interaktionen unterstützt
3. Mehr Flexibilität bezüglich des hier Daten-Modell genannten Domänen-Modells, das eine Vielfalt an Datentypen unterstützt
4. Unterstützung für aktuelle dynamische Technologien, wie z. B. *Asynchronous JavaScript and XML* (AJAX)[149]
5. Optimierung und Redundanz-Vermeidung in den Beschreibungssprachen für AUI und CUI, um kompaktere und lesbarere Spezifikationen zu erhalten.

Zusätzlich sind die Transformationsregeln in MARIAE nicht mehr, wie in TERESA, fest einprogrammiert, sondern lassen sich durch Konfiguration an die aktuellen Bedürfnisse anpassen [65].

---

[148] Siehe http://giove.isti.cnr.it/tools/MARIAE/download

[149] Konzept zur asynchronen Datenübertragung zwischen Browser und Server. Unter anderem wird die Möglichkeit geboten, eine Webseite partiell zu verändern, d.h. ohne sie gänzlich neu zu laden.

MARIAE unterstützt den Entwurf und die Entwicklung von interaktiven Anwendungen für mehrere Zielplattformen, die auf *Web Services* basieren. Gewöhnlicher Weise werden diese nicht im Rahmen der Entwicklung der interaktiven Applikation erstellt, sondern es wird auf bereits vorhandene *Web Services* zurückgegriffen. Dieser Tatsache wird auch im Entwurfs- und Entwicklungsprozess Rechnung getragen, der eine Mischung aus Top-down- und Bottom-up-Ansatz darstellt [137]. MARIAE folgt vollständig den Vorgaben des CRF (siehe Kapitel 2.1.3) und unterstützt somit die Abstraktionsgrade der Modell-Ebene, sowie der abstrakten, konkreten und finalen Benutzerschnittstelle [136]. Dies impliziert ein Top-down-Vorgehen im Entwicklungsprozess. Die Analyse und Planung der Verwendung bereits existierender *Web Services* stellt den genannten Bottom-up-Anteil dar [137].

Zunächst wird das Aufgaben-Modell der interaktiven Applikation unter Verwendung der ConcurTaskTrees-Notation erstellt. Im nachfolgenden Schritt werden die Verbindungen zwischen dem Aufgaben-Modell und den zuvor analysierten und ausgewählten *Web Services* hergestellt. Diese sind mittels der *Web Service Description Language*[150] (WDSL) beschrieben und verfügen über sogenannte *Annotations*[151], die unter anderem Hinweise zur späteren Darstellung in der Benutzerschnittstelle beinhalten. *Web Services* beziehen sich stets auf System-Tasks und deren Einbindung kann eine entsprechende Verfeinerung des Aufgaben-Modells notwendig machen. Das so entstandene angereicherte Task-Modell bildet die Grundlage für die anschließende Generierung der abstrakten Schnittstellenbeschreibung [136].

Das abstrakte User Interface setzt sich bei MARIA aus einer oder mehreren Präsentationen, einem Daten-Modell und einer Anzahl von externen Funktionen zusammen. Jede Präsentation besteht wiederum aus [138]:

1. Interface-Elementen, auch Interaktoren genannt
2. Kompositionen von Interaktoren
3. Einem Dialog-Modell, welches das dynamische Verhalten der Elemente beschreibt
4. Verbindungen, die anzeigen, wenn ein Wechsel zwischen Präsentation stattfinden soll

Die Transformation des angereicherten Task-Modells zum abstrakten Interface wird vornehmlich durch die hierarchische Struktur des Aufgaben-Modells, die Aufgabentypen, die temporalen Beziehungen zwischen den einzelnen Teilaufgaben und die in den *Annotations* enthalten Informationen beeinflusst. Zudem werden sogenannte *Presentation Task Sets* (PTS) identifiziert, die aus der Menge von zu einer Präsentation gehörenden elementaren Aufgaben bestehen. Sie zeichnen sich dadurch aus, dass sie während eines bestimmten Zeitraums gleichzeitig aktiviert sind [138].

In einem weiteren Schritt wird die abstrakte in eine plattformabhängige, konkrete Schnittstellenbeschreibung überführt. Diese Transformation kann als eine Verfeinerung des abstrakten Modells verstanden werden und umfasst hauptsächlich die Ersetzung abstrakter durch konkrete Interaktionsobjekte, die aus mehreren möglichen Repräsentationen ausgewählt werden. Die Transformationen werden mittels der *Extensible Stylesheet*

[150] Durch das *W3C* standardisierte plattform-, programmiersprachen- und zugriffsprotokollunabhängige Beschreibungssprache für Web Services, siehe http://www.w3.org/TR/wsdl

[151] Engl.: Anmerkungen, Erläuterungen

Language Transformation[152] (XSLT) spezifiziert. Die Beschreibungen der abstrakten und konkreten Benutzerschnittstellen, die auch die Informationen hinsichtlich der Dialog-, Daten- und Event-Modelle umfassen, erfolgen mithilfe von MARIA XML (siehe Kapitel 2.4.1.8). Letztlich wird noch die konkrete Beschreibung automatisch in eine Zielsprache wie beispielsweise XHTML umgesetzt. Auch diese Transformation erfolgt auf Basis von XSLT [138].

Hinsichtlich der Werkzeugunterstützung ist MARIAE reichhaltig ausgestattet und erlaubt durch entsprechende Editoren manuelle Eingriffe in die Modelle der verschiedenen Abstraktionsebenen und Transformationsspezifikationen. Darüber hinaus steht eine Vorschaufunktion für die finale Benutzerschnittstelle (FUI) zur Verfügung [137].

2.5.2. Vergleich und Bewertung

Es wurden bereits Studien zu Pattern-Werkzeugen durchgeführt und veröffentlicht, beispielswiese [18], [139] und [140]. Diese stammen jedoch aus den Jahren 1992, 1996 bzw. 2001. Deshalb sind dort einige neuere, sehr interessante Ansätze nicht enthalten. Nichtsdestotrotz lieferten diese Dokumente wertvolle Informationen, insbesondere hinsichtlich der zu untersuchenden Eigenschaften der modellbasierten Entwicklungsumgebungen.

Alle zuvor beschriebenen MB-UIDE wurden auf folgende Merkmale untersucht:

1. Name der MB-UIDE
2. Urheber
3. Datum des Entwicklungsbeginns bzw. der Erstveröffentlichung
4. Datum der Freigabe der aktuellsten Version bzw. der aktuellsten Publikation
5. Genereller Funktionalitätsumfang:
   a. UI-Modellierung
   b. UI-Generierung
   c. UI-Laufzeitumgebung
6. Unterstützung der im CAMELEON Reference Framework (siehe Kapitel 2.1.3) definierten Abstraktionsgrade „Modell-Ebene“, *Abstract User Interface* (AUI), *Concrete User Interface* (CUI), *Final User Interface* (FUI)
7. Art und Anzahl der verwendeten Modelle
8. Verwendete Modell-Notationen bzw. UIDL
9. Zielsetzung im Sinne einer Entwicklung von Benutzeroberflächen für mehrere:
   a. Endgeräte (engl. multi-device)
   b. Plattformen bzw. Programmiersprachen und Widget-Sets (engl. multi-platform)
   c. Benutzer (engl. multi-user)
   d. Nutzungsumgebungen (engl. multi-environment)
10. Unterstützende Werkzeuge
11. Unterstützte Ziel-Programmiersprachen
12. Unterstützte Endgeräte bzw. Ziel-Plattformen

[152] XSLT ist eine durch das *W3C* standardisierte Beschreibungssprache für die Transformation von XML-Dokumenten in andere XML-Dokumente, siehe http://www.w3.org/TR/xslt

13. Verfügbarkeit des MB-UIDE:
    a. Im Sinne der grundsätzlichen Existenz einer ablauffähigen Version
    b. Im Sinne eines tatsächlich möglichen Zugriffs auf das MB-UIDE
    c. Im Sinne von praktischen Anwendungsbeispielen
14. Art und Anzahl der vorhandenen Quellen und Dokumentationen

Ein Überblick über die im Literaturreview berücksichtigten und ausgewerteten Quellen wird in Tabelle 3 gegeben.

Tabelle 3 IM LITERATURREVIEW BERÜCKSICHTIGTE QUELLEN ZU DEN MB-UIDE

MB-UIDE	Quellen
UIDE	[79] [80] [81] [82] [83] [84] [85] [86] [87]
ITS	[88] [89] [90]
HUMANOID	[91] [92] [93] [94]
ADEPT	[95] [96] [97]
GENIUS	[98]
TRIDENT	[23] [24] [99] [100] [101]
AME	[102] [103] [104]
MASTERMIND	[80] [106] [107] [108] [109] [110]
JANUS	[111] [112] [113] [114]
FUSE	[115] [116] [117]
MECANO	[118] [119]
TADEUS (Rostock)	[120] [121] [22] [122] [21]
MOBI-D	[119] [123] [124] [125]
TEALLACH	[126]
TADEUS (Linz)	[127] [128]
TERESA	[58] [62] [73] [74] [141]
SUPPLE / SUPPLE++	[129] [130] [131] [132] [133] [134] [135]
MARIAE	[60] [64] [65] [76] [136] [137] [138] [142]

Zusammenfassend kann festgestellt werden, dass alle untersuchten MB-UIDE als wertvolle Beiträge zum Thema der modellgetriebenen Entwicklung von Benutzeroberflächen angesehen werden können. Keiner dieser Ansätze greift jedoch die Idee der Kombination von modellgetriebenen und musterbasierten Entwicklungsmethoden auf.

UIDE wurde im Lauf der Zeit in mehreren Generationen weiterentwickelt. In späteren Versionen findet die Modellierung der eigentlichen Anwendung und der Benutzerschnittstelle in separaten Modellen, den Anwendungs- und dem Interface-Modell, statt. Diese bestehen jeweils aus einem Daten- und einem Kontroll-Teilmodell, wobei letzteres die auf den definierten Datenobjekten ausführbaren Aktionen beschreibt. Somit kann zwischen den Datenobjekten und Aktionen auf der Anwendungsebene und jenen in der Benutzeroberfläche unterschieden werden. Die in den Modellen enthaltenen Informationen werden von der Laufzeitumgebung dazu genutzt, um eine ablauffähige Benutzeroberfläche zu erzeugen. Um adaptive Benutzeroberflächen zu ermöglichen, werden auch Benutzer-Modelle unterstützt.

Diese werden nicht zur Designzeit spezifiziert, sondern es erfolgt eine benutzerbezogene Protokollierung zur Laufzeit, um dynamische Änderungen der Benutzeroberfläche herzuleiten.

ITS verfolgt eine strikte Trennung von Anwendungs-Inhalten und deren Präsentation. Die Modellierung erfolgt anhand eines Daten- und eines Dialog-Modells. Aus den in diesen Modellen enthaltenen Informationen wird unter Zuhilfenahme eines ausgefeilten regelbasierten Ansatzes eine Repräsentation der Benutzerschnittstelle generiert, die schließlich von der in der Entwicklungsumgebung entahltenen Laufzeitumgebung ausgeführt werden kann.

Bei HUMANOID wird ein deklaratives Anwendungsmodell erstellt, in dem die Anwendungsobjekte, deren äußeres Erscheinungsbild, die möglichen Benutzeraktionen, die eventuell nach Benutzeraktionen automatisch durchzuführende Aktionen und die Dialogabfolge spezifiziert werden. Die Modellierung erfolgt inkrementell, d.h. das Modell wird grob erstellt und dann Schritt für Schritt verfeinert. Hierbei kommen sogenannte Templates zum Einsatz, die zeigen, wie komplexe Objekte in kleinere Einheiten zerlegt dargestellt werden können. Das fertige Modell wird von der in der Umgebung enthaltenen Laufzeitumgebung ausgeführt. Der Einsatz von Default-Werten ermöglicht die Ausführung von noch nicht vollständig spezifizierten Modellen.

Bei ADEPT wird auf Basis eines Aufgaben- und eines Benutzer-Modells ein sogenanntes Interface-Modell generiert. Dies ist ein zunächst abstraktes Modell der Benutzerschnittstelle, aus dem durch Verfeinerungen ein konkretes Modell entsteht. Das abstrakte Modell umfaßt abstrakte Interaktionsobjekte und die Beschreibung der Dialogstruktur. Das konkrete Modell beinhaltet Beschreibungen der konkreten Interaktionsobjekte einschließlich deren Verhalten und Layout, ist aber nach wie vor plattformunabhängig. Durch dessen Transformation entsteht abschließend die endgültige Benutzerschnittstelle.

Bei GENIUS werden ausgehend von einem existierenden Datenmodell sogenannte Views definiert. Die Views stellen logische Beschreibungen der Anwendungsfenster dar und umfassen Teile des Datenmodells. Aus ihnen wird durch Anwendung expliziter Design-Regeln das statische Layout der Benutzerschnittstelle erzeugt. Die Modellierung der dynamischen Aspekte erfolgt mittels Dialog-Netzen. Die Generierung wird durch eine Wissensbasis, die abstrakte Interaktionsobjekttypen und Designregeln zur Auswahl geeigneten Interaktionsobjekte und deren Anordnung enthält, unterstützt.

TRIDENT verfolgt das Ziel, Benutzerschnittstellen für hochgradig interaktive Datenbank-Anwendungen zu generieren. Die Modellierung erfolgt in Form eines Aufgaben- und eines Daten- und Funktions-Modells, aus denen ein sogenannter Activity-Chaining-Graph (ACG) abgeleitet wird. Dieser spezifiziert die jeweiligen Eingabe- und Ausgabedaten aller vorhandenen Funktionen sowie deren Relevanz für die Benutzeroberfläche. Die Dialogstruktur wird durch eine rekursive Zerlegung des ACG in Präsentationseinheiten erzeugt. Danach werden geeignete abstrakte Interaktionsobjekte ausgewählt, nach und nach durch konkrete Interaktionsobjekte ersetzt und schließlich in einem Layout platziert.

Mit AME wird eine enge Integration eines objekt-orientierten Softwareentwicklungs-Prozesses mit der Modellierung und dem Entwurf von Benutzeroberflächen verfolgt. Der Ausgangspunkt ist ein Domänen-Modell in Form eines OOA-Modells, das die Domänenobjekte sowie deren Attribute, Methoden und Objektbeziehungen enthält. Dieses wird anschließend zu einem OOD-Modell erweitert, das die Dialog- und Navigationsstrukturen beschreibt. Die Auswahl von abstrakten und kontrekten Interaktionsobjekten erfolgt regelbasiert. Alle zur

Erzeugung der Benutzerschnittstelle notwendigen Schritte können automatisiert ablaufen, aber jederzeit auch durch manuelle Eingriffe beeinflusst werden.

MASTERMIND ist die Zusammenführung und Weiterentwicklung der beiden modellbasierten UI-Entwicklungsumgebungen HUMANOID und UIDE. Die Modellierung erfolgt auf drei Abstraktionsebenen. Zunächst werden ein Aufgaben- und ein Applikations-Modell auf der semantischen Ebene erstellt. Auf dieser Basis wird dann ein Dialog-Modell auf der syntaktischen Ebene und anschließend Präsentations- und Interaktions-Modelle auf der lexikalischen Ebene spezifiziert. Abschließend erfolgt die Erzeugung von Quellcode für die Zielplattform durch einen Generator.

Ähnlich wie bei AME ist auch bei JANUS ein OOA-Domänen-Modell der Ausgangpunkt für die Code-Generierung. Das Modell wird von zwei verschiedenen Generatoren verarbeitet. Einerseits ist dies ein GUI-Generator, der Dialogstrukturen, Fenster und Interaktions-Elemente erzeugt. Für die Layout-Generierung steht ein automatisiertes Werkzeug zur Verfügung. Andererseits erstellt ein Applikations-Generator automatisiert Code-Fragmente der Anwendungslogik, die Anbindung an die Datenbank und das Datenbank-Schema. Ähnlich wie bei ITS verwendet JANUS bei der GUI-Erzeugung Voreinstellungen, die aber durch manuelle Eingriffe abgeändert werden können.

Die Erzeugung von Benutzerschnittstellen erfolgt bei FUSE im Rahmen eines Entwicklungsprozesses mit den drei Phasen Analyse, Design und Evaluation. Die eigentliche Implementierung erfolgt durch automatische Generierung. In allen Phasen stehen unterstützende Werkzeuge zur Verfügung. Im ersten Schritt werden Aufgaben-, Domänen- und Benutzer-Modelle erstellt, auf deren Basis dann eine von der endgültigen Präsentation unabhängige, logische Benutzerschnittstelle abgeleitet wird. Aus dieser wird schließlich eine ausführbare UI-Spezifikation erzeugt, bei der zur Laufzeit zwischen unterschiedlichen Layout-Stilen umgeschaltet werden kann.

Bei MECANO wird die Benutzerschnittstelle aus einem zum Domänen-Modell erweiterten Daten-Modell generiert. Zunächst wird mittels der Informationen aus diesem Modell eine High-Level-Dialog-Spezifikation erzeugt und anschließend das UI-Layout auf Basis von Design-Guidelines festgelegt. Danach erfolgt die Generierung eines Low-Level-Dialogs, der die dynamischen Aspekte der späteren Bedienungsoberfläche spezifiziert. Schließlich erfolgt noch die Feinanpassung der User-Schnittstelle gegebenenfalls unter Einbeziehung von Benutzern. Die Weiterentwicklung von MECANO erfolgte unter dem Namen MOBI-D.

Die Modellierung der Benutzerschnittstelle erfolgt bei TADEUS (Rostock) auf Basis von Aufgaben-, Domänen-, Benutzer- und Dialog-Modellen. Im Mittelpunkt steht die explizite Dialogmodellierung mittels Dialog-Graphen, bei der sowohl statische als auch dynamische Aspekte der Mensch-Maschine-Kommunikation berücksichtigt werden. Aus den Informationen der Eingangsmodelle wird eine rudimentäre Version des Dialog-Modells generiert und dann Schritt für Schritt verfeinert. Schließlich erfolgt die automatisierte Erzeugung der finalen Benutzeroberfläche mithilfe einer Wissensbasis für Software-Ergonomie.

MOBI-D ist die Weiterentwicklung von MECANO. Der Modellierungsansatz wurde dabei um User-Task-, Domänen-, Präsentations-, Dialog-, und User-Modelle erweitert. Die Beziehungen zwischen Elementen dieser Modelle werden im ebenfalls neu eingeführten Design-Modell beschrieben. Zunächst werden die User-Task- und Domänen-Modelle und optional das User-Modell erstellt und die Beziehungen zwischen diesen Modellen festgelegt. Im Unterschied zu MECANO werden die anschließenden Design-Entscheidungen nicht

vollständig automatisiert getroffen, sondern sie werden automatisch berechnet und priorisiert, können aber jederzeit beliebig geändert oder verworfen werden.

Bei TEALLACH erfolgt die Generierung von Benutzerschnittstellen auf Basis von Domänen-, Aufgaben- und Präsentations-Modellen. Es wird davon ausgegangen, dass die zugrundeliegende Anwendung, für das das User-Interface entwickelt werden soll, bereits vorhanden und somit das Domänenmodell fest vorgegeben ist. Das Präsentations-Modell beschreibt sowohl das Erscheinungsbild als auch das Verhalten der Benutzeroberfläche und bildet die Abstraktionsebenen der abstrakten und konkreten Benutzerschnittstelle ab. Im Rahmen des Modellierungsprozesses werden zudem Beziehungen zwischen den genannten Modellen definiert.

Bei TADEUS (Linz) erfolgt die Modellierung anhand eines sogenannten Anwendungs-Modells, das aus den vier Teilen Aufgaben-, Benutzer-, Domänen- und Interaktions-Modell besteht. Das Design einer Benutzerschnittstelle wird als kontinuierlicher Wechsel zwischen diesen Perspektiven mit einhergehender schrittweiser Verfeinerung der betreffenden Modelle verstanden. Das Interaktions-Modell legt hierbei sowohl die Anordnung und den Darstellungsstil von generischen oder plattformspezifischen Interaktionselementen als auch deren dynamisches Verhalten fest.

Die Basis der UI-Generierung bildet bei TERESA ein angereichertes Task-Modell, das neben den Benutzeraufgaben auch die zu deren Ausführung benötigten Datenobjekte und deren Beziehungen zueinander beinhaltet. Durch Filterung und Verfeinerung werden daraus plattformspezifische System-Task-Modelle erstellt, aus denen abstrakte Beschreibungen der Benutzerschnittstelle abgeleitet werden. Danach erfolgt die Ersetzung der darin enthaltenen abstrakten Interaktionsobjekte durch die auf der Zielplattform zur Verfügung stehenden konkreten Interaktionsobjekte. Auf dieser Basis wird dann das endgültige User-Interface generiert. Die Weiterentwicklung von TERESA erfolgte unter dem Namen MARIAE.

Bei SUPPLE erfolgt die Modellierung anhand von User- und Device-Modellen sowie sogenannter Interface-Spezifikationen. Letztere sind durch einen UI-Designer vorgegebene Mengen von abstrakten Schnittstellen-Elementen, die jeweils durch ihren Datentyp und die bei ihrer Verwendung geltenden Einschränkungen definiert werden. Die Erzeugung und Adaption von Benutzeroberflächen wird auf Basis von Lösungen zu Optimierungsproblemen behandelt und erfolgt zur Laufzeit. Es wird nach Darstellungen gesucht, bei denen durch das verwendete Endgerät entstehende Einschränkungen berücksichtigt werden und den erwarteten Aufwand für die Ausführung der Benutzeraktionen minimiert. SUPPLE++ nutzt den SUPPLE-Ansatz zur automatischen Generierung und Anpassung von User-Interfaces für Benutzer mit Sehbehinderungen und/oder eingeschränkten motorischen Fähigkeiten.

MARIAE ist die Weiterentwicklung der modellbasierten UI-Entwicklungsumgebung TERESA. Ziel ist der Entwurf von interaktiven Anwendungen, die auf bereits existierenden Web-Services basieren und auf mehreren Zielplattformen zur Verfügung stehen. MARIAE ist gemäß des CAMELEON Reference Frameworks implementiert und unterstützt somit die Abstraktionsgrade der Modell-Ebene und der abstrakten, konkreten und finalen Benutzerschnittstelle. Ausgangsbasis ist ein Aufgaben-Modell, das um Web-Service-bezogene Datenaspekte erweitert wird. Danach erfolgen die Transformationen über AUI zu CUI und abschließend in die Sprache der Zielplattform. MARIAE wird aktuell noch weiterentwickelt.

Zusammenfassend lassen sich die untersuchten modellbasierten UI-Entwicklungsumgebungen in zwei grundsätzliche Kategorien einteilen. Einerseits sind dies diejenigen Ansätze, bei

denen die Modellierung im Wesentlichen anhand eines Daten-Modells erfolgt. Dieses kann aber durchaus um zusätzliche Informationen erweitert sein. Zu dieser Kategorie gehören UIDE, ITS, HUMANOID, GENIUS, AME, JANUS, MECANO und SUPPLE bzw. SUPPLE++. Andererseits gibt es Umgebungen, bei denen auf oberster Abstraktionsebene Aufgaben-Modelle zum Einsatz kommen. Hierzu zählen ADEPT, TRIDENT, MASTERMIND, FUSE, TADEUS (Rostock), MOBI-D, TEALLACH, TADEUS (Linz), TERESA und MARIAE.

Die im CAMELEON Reference Framework (siehe Kapitel 2.1.3) definierten Abstraktionsgrade „Modell-Ebene“, *Abstract User Interface* (AUI), *Concrete User Interface* (CUI), *Final User Interface* (FUI) werden von TRIDENT, AME, JANUS, TADEUS (Rostock), TERESA und MARIAE vollständig unterstützt. Eine Unterscheidung von AUI und CUI erfolgt mit Ausnahme von MASTERMIND bei allen untersuchten MB-UIDE.

Als lauffähige Versionen lagen ausschließlich TERESA und MARIAE, die frei im Internet erhältlich waren, vor.

Die vollständige Beschreibung der Untersuchungsergebnisse in Tabellenform befindet sich in Anhang B.

3. Musterbasierte Entwicklung

Wie bereits mehrfach erwähnt, befasst sich die vorliegende Dissertation mit dem Entwurf eines Frameworks zur Modellierung und Generierung von Benutzeroberflächen für interaktive Systeme. Die Strategie ist hierbei die Kombination von modellgetriebenen und musterbasierten Entwicklungsansätzen. Das folgende Kapitel vermittelt den Einstieg in die musterbasierte Software-Entwicklung. Es erklärt wesentliche Aspekte dieser Disziplin, die für das Verständnis der Funktionsweise des später beschriebenen Frameworks erforderlich sind und die explizit oder implizit in dessen Entwurf eingeflossen sind.

Dazu werden zunächst in Kapitel 3.1 erforderliche Grundlagen beschrieben. Dies beinhaltet wichtige Begriffsdefinitionen sowie Informationen über die chronologische Weiterentwicklung von Patterns, existierende Klassifizierungsansätze und den zentralen Aspekt der Beziehungen zwischen den Mustern. Kapitel 3.2 befasst sich mit insgesamt sechs existierenden formalen Beschreibungssprachen für Patterns, die im Detail erklärt und anschließend bewertet und verglichen werden. Anschließend erfolgt in Kapitel 3.3 die Beschreibung und Analyse von fünf bestehenden Pattern-Sammlungen. Schließlich widmet sich Kapitel 3.4 einer Anzahl von vierzehn Pattern-Werkzeugen, die detailliert beschrieben und mittels einer speziell dafür definierten Metrik analysiert und bewertet werden.

3.1. Grundlagen

Das folgende Kapitel führt in die Grundlagen der musterbasierten Softwareentwicklung ein. Nach der Definition von Grundbegriffen folgt ein Überblick über die Chronologie der Entwicklung und Verwendung von Mustern. Danach werden Klassifikationsansätze für Patterns dargestellt und abschließend wird auf eine der wichtigsten Eigenschaften von Mustern, die Pattern-Beziehungen, eingegangen.

3.1.1. Begriffsdefinition

3.1.1.1. Pattern

Muster (engl. patterns) dienen dazu, Expertenwissen zu sammeln, zu dokumentieren und für Andere verfügbar zu machen. Ferner übernehmen sie in der Praxis die Funktion einer *Lingua Franca* und tragen somit zur Überwindung von fachlichen und kulturellen Hindernissen bei der Kommunikation zwischen verschiedenen Beteiligten bei [143].

Christopher Alexander definiert den Begriff eines Musters in seinem Buch *The Timeless Way of Building* wie folgt [144]:

Ein Muster ist eine dreiteilige Regel, die eine Beziehung zwischen einem bestimmten Kontext, einem Problem und einer Lösung ausdrückt. Einerseits ist ein Muster eine Sache, die auf der Welt passiert und andererseits gleichzeitig die Gesetzmäßigkeit, die aussagt, wie diese Sache erschaffen werden kann und wann dies erfolgen muss [144].

Hierbei beschreibt der Kontext eine wiederkehrende Anzahl von Situationen, in denen das Pattern angewendet werden kann. Das Problem bezieht sich auf eine Menge von Einflüssen (engl. forces) in Form von Zielen (engl. goals) und Beschränkungen (engl. constraints), die im gegebenen Kontext auftreten. Grundsätzlich beschreibt es, wann das Muster angewendet werden kann. Die Lösung beschreibt, wie die Ziele erreicht bzw. die Einschränkungen beseitigt werden können [143].

3.1.1.2. HCI-Design-Pattern

Im Kontext der Mensch-Computer-Interaktion (engl. human-comuter interaction, HCI) beschreibt ein Entwurfs-Muster (engl. design pattern) ein wiederkehrendes Problem zusammen mit einer bewährten Lösung. Ein Pattern liegt in einem wohldefinierten Beschreibungsformat vor, das von den individuellen Präferenzen des jeweiligen Autors abhängt. Entwurfs-Muster können untereinander in Beziehung stehen [145] [146].

In [147] findet sich folgende Definition:

Entwurfsmuster vermitteln Einblicke in Design-Probleme und erfassen das Wesentliche dieser Probleme und deren Lösungen in kompakter Form. Sie umfassen jeweils eine detaillierte Beschreibung des vorliegenden Problems, Begründungen für die vorgeschlagene Lösung und mögliche Konflikte, die bei der Umsetzung der Lösung entstehen können [147].

Gemäß [143] ist ein HCI-Design-Pattern:

- Eine Menge von Regeln, die dazu dienen, Dinge auszuführen oder ganz oder teilweise zu erzeugen
- Eine allgemeine, wiederholbare Interaktionstechnik für ein häufig auftretendes Benutzer-Problem
- Eine invariante Lösung zu einem wiederkehrenden Design-Problem in einem bestimmten Kontext
- Eine allgemein wiederholbare Lösung zu einem verbreitet auftretenden Usability-Problem beim Interaktions- oder Benutzerschnittstellen-Design
- Eine Lösung zu einem Usability-Problem, das in mehreren Kontexten auftritt
- Eine erfolgreiche, bewährte HCI-Design-Lösung hinsichtlich Entwurf, Entwicklung, Bewertung und/oder Benutzung von interaktiven Systemen

Einerseits müssen HCI-Entwurfsmuster konkret genug formuliert sein, damit sie als fundierte Lösungen zu Design-Problemen in der Praxis unmittelbar umgesetzt werden können. Andererseits müssen sie aber auch abstrakt genug sein, um an verschiedene Situationen angepasst werden zu können [143].

HCI-Design-Patterns werden alternativ auch User-Interface-Design-Patterns, Interaction-Patterns, Usability-Patterns oder HCI-Patterns genannt [143]. Im Folgenden werden sie als Design-Patterns bzw. Entwurfsmuster oder abkürzend als Patterns bzw. Muster bezeichnet.

3.1.1.3. Antipattern

Ein Antipattern ist als Gegenteil eines Patterns zu verstehen [148]. Es beschreibt eine häufig verwendete Lösung zu einem Problem, die unwirksam ist oder vernehmlich negative Konsequenzen verursacht. Es handelt sich um eine Lösung, die sich vordergründig als gute Idee präsentiert, aber in der Realität fehlschlägt, insbesondere wenn sie in einem falschen

Kontext umgesetzt wird [149] [150]. Es wird zwischen einfachen Antipatterns (engl. simple anti-patterns) und sogenannten Verbesserungs-Antipatterns bzw. Verbesserungspatterns (engl. amelioration anti-patterns bzw. amelioration patterns) unterschieden. Ein einfaches Antipattern beschreibt, wie man von einem Problem zu einer schlechten Lösung kommt, d.h. die negativen Aspekte stehen im Fokus. Verbesserungspatterns beschreiben ebenfalls den Weg von einem Problem zu einer schlechten Lösung, zeigen aber zudem auf, wie man von der schlechten zu einer guten Lösung gelangt. Darüber hinaus wird beschrieben, warum die schlechte Lösung attraktiv erscheint, warum sie sich letztlich als unbrauchbar herausstellt und welche Lösungen bzw. welche Patterns stattdessen geeignet sind [150] [148].

3.1.1.4. Generative und nicht-generative Patterns

Muster, die in einem bereits implementierten System zu beobachten sind, werden als nicht-generativ bezeichnet. Sie sind deskriptiv und passiv [151] [152], da die Begründung für deren Anwendung nicht vorliegt [151] und eine gesicherte Reproduktion der beobachteten Phänomene nicht möglich ist [152] [153]. Generative Muster sind hingegen dazu gedacht, Systemarchitekturen zu gestalten und Systeme ganz oder teilweise zu generieren [151] [152]. Sie sind präskriptiv und aktiv [151].

3.1.2. Zeitliche Entwicklung

Ihren Ursprung haben die Entwurfsmuster in der zweiten Hälfte der 1970er Jahre, als Christopher Alexander entsprechende Konzepte zur Ermöglichung der Wiederverwendung von Design-Know-how im Bereich des Gebäude- und Städtebaus entwickelt und diese in einer Reihe von Büchern veröffentlicht [144] [154] [155]. Im Jahr 1986 wird der Begriff des Musters in einem Beitag von Edwin Hutchins, James Hollan und Donald Norman im Buch mit dem Titel *User Centered System Design – New Perspectives on Human-Computer Interaction* [156] erstmals in Zusammenhang mit Benutzerschnittstellen und HCI gebracht [146]. Ein Jahr später führen Kent Beck und Ward Cunningham Patterns in der objektorientierten Programmierung ein und berichten von einem erfolgreichen Entwurf einer Benutzeroberfläche mithilfe von fünf Mustern [157]. Breites öffentliches Interesse erreichten Entwurfsmuster dann im Jahr 1995 mit der Veröffentlichung des Buchs *Design Patterns: Elements of Reusable Object-Oriented Software* von Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, bekannt auch als sogenannte Viererbande (engl. Gang of Four, GoF) [31]. Von da an entstand eine Vielzahl weiterer Veröffentlichungen zu Pattern-Konzepten sowohl im Bereich der HCI, beispielsweise [158], als auch in anderen Domänen. Dazu zählen unter anderem *Usability Engineering* (UE) [159], *User Experience* (UX) [160], Workflow-Systeme [161], Aufgaben-Modellierung [162] und Applikations-Sicherheit [163].

3.1.3. Klassifikation von Patterns

Grundsätzlich können Patterns in die Klassen der generativen und nicht-generativen Patterns eingeteilt werden [150]. Hierbei sind generative Patterns stets zu bevorzugen, da sie nicht nur die Charakteristika guter Systeme beschreiben, sondern auch aufzeigen, wie diese zu erzeugen sind [153].

In [153] und [164] erfolgt eine Einteilung in Architektur-Muster (engl. architectural patterns), Entwurfsmuster (engl. design patterns) und Idiome (engl. idioms). Ein Architektur-Muster beschreibt hierbei eine grundlegende, strukturelle Organisation von Softwaresystemen. Es liefert eine Menge von vordefinierten Subsystemen, beschreibt deren Zuständigkeiten und enthält Regeln und Richtlinien hinsichtlich der Beziehungen zwischen diesen Subsystemen. Entwurfsmuster enthalten Informationen bezüglich der Verfeinerung der Subsysteme, deren Komponenten oder Beziehungen zueinander. Sie beschreiben häufig wiederkehrende Strukturen kommunizierender Komponenten, die ein generelles Design-Problem in einem bestimmten Kontext lösen. Idiome beziehen sich auf eine Programmiersprache. Sie zeigen auf, wie bestimmte Aspekte von Komponenten oder deren Beziehungen zueinander mit den zur Verfügung stehenden Möglichkeiten implementiert werden können.

Diese drei Klassen von Patterns unterscheiden sich hinsichtlich Abstraktionsgrad und Detailliertheit. Architekturmuster machen strategische Vorgaben auf oberster Ebene und beziehen sich auf große Komponenten und globale Eigenschaften und Mechanismen von Systemen. Sie haben weitreichende Auswirkungen und beeinflussen die gesamthafte Struktur und Organisation eines Systems. Entwurfsmuster bewegen sich auf der mittleren Abstraktionsebene und beziehen sich auf Struktur und Verhalten von Teilen eines Systems und deren Beziehungen untereinander. Sie haben zwar keinen Einfluss auf die Gesamtarchitektur, spezifizieren aber sogenannte Mikroarchitekturen von Subsystemen oder Komponenten. Idiome halten schließlich Details auf der untersten, von der jeweiligen Programmiertechnik abhängigen Ebene hinsichtlich der Struktur und des Verhaltens von Komponenten bereit [153] [164].

Eine ähnliche Klassifikation von Mustern ist in [165] beschrieben, orientiert sich jedoch an den Softwareentwicklungsphasen der Analyse, des Entwurfs und der Implementierung [153] sowie den dabei jeweils verwendeten Modellen [165]. Auf dieser Basis wird zwischen konzeptuellen Mustern (engl. conceptual patterns), Enwurfsmustern (engl. design patterns) und Programmiermustern (engl. programming patterns) unterschieden.

Konzeptuelle Muster werden in der Terminologie und anhand der existierenden Konzepte der betreffenden Anwendungsdomäne beschrieben [153] [165]. Sie helfen dabei, die Domäne und die in dieser existierenden Aufgabe zu verstehen. Es ist anzumerken, dass konzeptuelle Patterns keine allgemeine Gültigkeit besitzen, sondern auf die betreffende Domäne beschränkt sind [165]. Entwurfsmuster sind hierbei mittels der zur Verfügung stehenden Softwaredesign-Konstrukte zu beschreiben, also beispielsweise durch Objekte, Klassen, Vererbung und dergleichen. Sie verfeinern oder vervollständigen den durch die konzeptuellen Patterns skizzierten Gegenstand. Die Programmiermuster beschreiben die Implementierungsdetails auf Basis der durch die jeweilige Programmiersprache vorgegebenen Konstrukte [153] [165] und entsprechen somit den oben definierten Idiomen [153].

3.1.4. Beziehungen zwischen Patterns

Eine Steigerung ihres Mehrwerts erfahren Patterns durch die Tatsache, dass sie nicht isoliert nebeneinander, sondern in Beziehungen zueinander stehen. Aus diesem Grund ist es notwendig, diese bestehenden Relationen zu erfassen und formal zu beschreiben (siehe Kapitel 3.2). Besondere Bedeutung kommt hierbei auch den sogenannten Pattern-Sprachen (siehe Kapitel 3.3.1.2) zu, die Mengen von in bestimmten Beziehungen zueinander stehen-

den Mustern zusammenfassen und somit praktisch ein Netzwerk von miteinander verbundenen Patterns darstellen [166].

In [166] werden Mustern die aus der Objektorientierung bekannten Beziehungen der Aggregation, Spezialisierung und Assoziation zugeschrieben. Hierbei bezeichnet die Aggregation eine „has-a"-, die Spezialisierung eine „is-a"- und die Assoziation eine „related-to"-Beziehung.

Darüber hinaus wurden weitere Studien zu Beziehungen von Mustern veröffentlicht, beispielsweise [164], [167], [168], [169] und [170]. Die dort beschriebenen Relationstypen sind jeweils mit kurzen Erklärungen und den Literaturquellen in Tabelle 4 zusammengefasst.

Tabelle 4 TYPEN VON BEZIEHUNGEN ZWISCHEN PATTERNS

Beziehung	Beschreibung	Quellen
Similar-to	Pattern A ist ähnlich zu Pattern B	[167] [168] [170] [171]
Uses	Pattern A verwendet Pattern B innerhalb seiner Lösung	[167] [168] [169] [170] [171]
Refines	Pattern B ist eine spezifischere Lösung als Pattern A	[164] [167] [168] [169]
Combinable-with	Pattern A und Pattern B können kombiniert werden	[164] [167] [168] [170]
Variation-of	Pattern B ist Pattern A mit Änderungen hinsichtlich der Lösung (in Pattern B)	[164] [167] [168]
Provides-context	Pattern A und Pattern B können sequenziell angewendet werden	[167] [169]
Conflicts-with	Die gleichzeitige Anwendung von Pattern A und Pattern B ist ausgeschlossen	[168] [171]
Is-neighbour	Pattern A und Pattern B gehören der gleichen Pattern-Kategorie an	[171]

3.2. Formale Beschreibung von Mustern

Das folgende Kapitel gibt einen Überblick über existierende Standardisierungsbemühungen hinsichtlich der formalen Beschreibung von Patterns. Die dazu verwendeten Schemata werden in der Literatur gelegentlich auch als Pattern-Templates bezeichnet, z. B. in [150]. In Unterkapitel 3.2.1 sind die im Rahmen dieser Dissertation im Detail untersuchten Beschreibungssprachen *PLML 1.1*, *PLML 1.2*, *PLMLx*, *XPLML*, *PCML* und *TPML* beschrieben. Abschließend befindet sich in Unterkapitel 3.2.2 ein Vergleich und die Bewertung dieser formalen Beschreibungsansätze. Die betreffenden Inhalte wurden bereits teilweise in [172] veröffentlicht

3.2.1. Untersuchte Pattern-Beschreibungssprachen

3.2.1.1. PLML Version 1.1

Im Rahmen der *CHI 2003 Conference on Human Factors in Computing Systems*[153] wurde der Workshop *Perspectives on HCI Patterns: Concepts and Tools* [154] abgehalten. Ziel dieser Veranstaltung war unter anderem die Definition einer einheitlichen Beschreibungssprache für HCI-Patterns und HCI-Pattern-Sprachen, die *Pattern Language Markup Language* (PLML[155]). PLML sollte die bis dato unterschiedlichen Ansätze zur Spezifikation von Mustern standardisieren und zugleich die automatisierte Verarbeitung der Patterns mit Hilfe von einheitlichen Werkzeugen ermöglichen. Die Teilnehmer einigten sich schließlich auf die PLML 1.1 genannte Sprache, deren Beschreibungselemente beziehungsweise –attribute in Tabelle 5 aufgelistet und kurz beschrieben werden [173].

Tabelle 5 PLML 1.1 BESCHREIBUNGSELEMENTE UND -ATTRIBUTE

Element/Attribut	Beschreibung
<patternID>	Eindeutiger Bezeichner des Musters
<name>	Name des Musters
<alias>	Alternative Namen des Musters
<illustration>	Gutes, verständliches Beispiel der Anwendung des Musters
<problem>	Beschreibung des zu lösenden Problems
<context>	Situationen , in der das Muster angewendet werden kann
<forces>	Beschreibung der Einflüsse in der Umgebung, die bei der Anwendung des Musters einwirken
<solution>	Beschreibung, wie das Problem zu lösen ist
<synopsis>	Zusammenfassung der Pattern-Beschreibung
<diagram>	Schematische Darstellung des Musters
<evidence> <example> <rationale>	Nachweis, dass es sich tatsächlich um ein Muster handelt, durch mindestens drei bekannte Anwendungen des Muster Diskussion und jegliche grundsätzliche Begründungen
<confidence>	Einschätzung der Wahrscheinlichkeit, dass das Muster tatsächlich eine invariante Lösung des gegebenen Problems darstellt
<literature>	Verweise auf diesbezügliche Dokumente und Veröffentlichungen
<implementation>	Code-Fragmente oder Details hinsichtlich der technischen Realisierung
<related-patterns>	Beziehungen zu anderen Mustern

[153] Fort Lauderdale, Florida, USA, http://www.chi2003.org/index.cgi
[154] Engl.: Perspektiven für HCI-Muster: Konzepte und Werkzeuge
[155] Ausgesprochen: „Pell-Mell“

Element/Attribut	Beschreibung
<pattern-link>	Verknüpfung von Mustern
<type>	Art der Beziehung
<patternID>	Bezeichner des betreffenden Musters
<collection>	Bezeichner der betreffenden Muster-Sammlung
<label>	Name der Pattern-Verknüpfung
<management>	Urheberschaft und Änderungskontrolle
<author>	Name des Autors
<credits>	Verdienste
<creation-date>	Datum der Erfassung
<last-modified>	Datum der letzten Änderung
<revision-number>	Versionsnummer der Pattern-Definition

3.2.1.2. PLML Version 1.2

Bei der Entwicklung des Werkzeugs *Management of User Interface Patterns* (MUIP, siehe Kapitel 3.4.2.4) zur Erstellung und Verwaltung von Mustern wurde PLML 1.2 definiert. Es wurde dabei beabsichtigt, die Organisation von Patterns und deren Inhalten effektiver zu gestalten. Die erste Veröffentlichung geht auf das Jahr 2006 zurück [174].

Die hauptsächlichen Erweiterungen betreffen die Elemente <forces>, <examples> und <management>. Bei <forces> werden nun die betreffenden Einflüsse mithilfe des Unterelements <force> einzeln repräsentiert und sind somit separat such- und wiederverwendbar. Gleiches gilt auch für das Element <example>, das nun die einzelnen Beispiele getrennt voneinander durch die Unterelemente <example-name>, <example-diagram>, <description> und <knownuses> spezifiziert. Das Element <management> erhielt ein zusätzliches Unterelement <change-log>, das dazu dient, die Änderungshistorie zu beschreiben. Hierfür hat <change-log> wiederum die Unterelemente <log-creation-date> und <log-content> erhalten. Darüber hinaus wurde das neue Attribut <collection> eingeführt, um die Zugehörigkeit des Musters zu einer Pattern-Sammlung zu definieren. Das Element <pattern-link> erhielt das zusätzliche Unterelement <revision-number>, damit nun auch Beziehungen zwischen verschiedenen Versionen von Mustern spezifiziert werden können. Ferner wurde die Struktur der Elemente <literature> durch die Unterelemente <workname> und <reference> und <implementation> durch <implementation-name>, <code> und <otherdetails> verfeinert [174].

Die Änderungen gegenüber PLML 1.1 und die Erklärungen zu den genannten Attributen, Elementen und Unterelementen sind in Tabelle 6 zusammengefasst.

Tabelle 6 ZUSÄTZLICHE ODER GEÄNDERTE BESCHREIBUNGSELEMENTE UND -ATTRIBUTE VON PLML 1.2 IM VERGLEICH ZU PLML 1.1

Element/Attribut	Beschreibung
<collection>	Name der Pattern-Sammlung, zu der das vorliegende Muster gehört.
<forces> <force>	Einflüsse (wie bei PLML 1.1), jedoch werden diese jeweils separat repräsentiert und sind somit einzeln such- und wiederverwendbar.

Element/Attribut	Beschreibung
<example>	Anwendungsbeispiele (wie PLML 1.1), jedoch weiter strukturiert und einzeln zugreifbar
<example-name>	Name des Beispiels
<example-diagram>	Schematische Darstellung des Beispiels
<description>	Beschreibung des Beispiels
<knownuses>	Bekannte Fälle, in denen das Muster eingesetzt wurde
<literature>	Literatur-Referenzen (wie PLML 1.1), jedoch weiter strukturiert und einzeln zugreifbar
<workname>	Name des betreffenden Literaturwerkes
<reference>	Referenz auf das Werk
<implementation>	Details der technischen Realisierung (wie PLML 1.1), jedoch weiter strukturiert und einzeln zugreifbar
<implementation-name>	Name der Implementierung
<code>	Code oder Code-Fragmente
<otherdetails>	Weitere Details der Implementierung
<pattern-link>	Verknüpfung von Mustern (wie PLML 1.1)
<type>	Art der Beziehung (wie PLML 1.1)
<patternID>	Bezeichner des betreffenden Musters (wie PLML 1.1)
<collection>	Bezeichner der betreffenden Muster-Sammlung (wie PLML 1.1)
<label>	Name der Pattern-Verknüpfung (wie PLML 1.1)
<revision-number>	Bezeichner der Überarbeitung, der ermöglicht, Beziehungen zwischen verschiedenen Versionen von Mustern zu definieren
<management>	Urheberschaft und Änderungskontrolle (wie PLML 1.1)
<author>	Name des Autors (wie PLML 1.1)
<credits>	Verdienste (wie PLML 1.1)
<creation-date>	Datum der Erfassung (wie PLML 1.1)
<last-modified>	Datum der letzten Änderung (wie PLML 1.1)
<revision-number>	Versionsnummer der Pattern-Definition (wie PLML 1.1)
<change-log>	Informationen zu am Pattern durchgeführten Änderungen
<log-creation-date>	Datum der Änderung
<log-content>	Beschreibung der Änderung

3.2.1.3. PLMLx

Die *Extended Pattern Language Markup Language* (PLMLx) wurde primär dazu entworfen, um Patterns in einem gut darstell- und lesbaren, artikelähnlichen Format beschreiben zu können. Hierzu wurden einige Änderungen an PLML 1.1 vorgenommen und insbesondere die verwendeten Datentypen so modifiziert, dass die Publishing-Software DocBook[156] optimal unterstützt wird [175].

PLMLx führt die drei neuen Beschreibungselemente <acknowledgements>, <resulting-context> und <organization> ein. Dabei erlaubt <acknowledgements> die Nennung von an der Entwicklung des Patterns maßgeblich beteiligten Personen und <resulting-context> beschreibt den Kontext, in dem man sich befindet, nachdem das Muster angewendet wurde. Dieser kann wiederum ein oder mehrere neue zu lösende Probleme einschließen.

[156] Siehe http://www.docbook.org

Das Element <organization> ist zur Spezifikation von organisatorischen Metadaten gedacht und besitzt die drei Unterelemente <category>, <collection> und <classification>. Mit <category> kann das Pattern einer frei definierbaren Kategorie zugeordnet werden, z. B. „HCI“ oder „Organizational“, während mit dem Element <collection> eine oder mehrere Mustersammlungen genannt werden können, in denen das Pattern enthalten ist. <classification> dient zur Klassifizierung des Patterns. Es gibt mehrere verschiedene Klassifizierungsansätze, z. B. nach GoF[157] oder POSA[158], die hierbei herangezogen werden können. Das Element wird aber zunächst als Platzhalter für eine spätere, standardisierte Taxonomie angesehen [175].

Grundlegend erweitert wurde das Element <management> durch die drei zusätzlichen Unterelemente <copyright>, <license> und <change-log>. Hierbei nimmt <copyright> Informationen zum Urheberrecht auf und <license> gibt Auskunft über die Lizenzbedingungen, unter denen das Pattern veröffentlicht wird. Dies wird mit den beiden Unterelementen <license-type> und <ulink> erreicht. Hierbei ist der Lizenztyp frei formulierbar, z. B. „GNU Free Documentation License“, „Common Documentation License“ oder „Open Publication License“, während <ulink> optional den Uniform Resource Locator[159] (URL) der Lizenzvereinbarung spezifiziert. Schließlich wurde das PLML 1.1-Element <evidence> entfernt und dessen beiden Unterelemente <example> und <rationale> nun als Elemente der ersten Ebene geführt [175].

Die Erweiterungen und Änderungen in PLMLx gegenüber PLML 1.1 und die Erklärungen zu den genannten Attributen, Elementen und Unterelementen sind in Tabelle 7 zusammengefasst.

Tabelle 7 ZUSÄTZLICHE ODER GEÄNDERTE BESCHREIBUNGSELEMENTE UND -ATTRIBUTE VON PLMLX IM VERGLEICH ZU PLML 1.1

Element/Attribut	Beschreibung
<acknowledgements>	Dank und Anerkennung für alle Beteiligten, die bei der Entwicklung des Patterns maßgeblich beigetragen haben.
<evidence> <example> <rationale>	Das Beschreibungselement <evidence> wurde gestrichen. Die beiden Unterelemente <example> und < rationale> werden in PLMLx als Elemente der ersten Ebene geführt.
<organization> <category> <collection> <classification>	Organisatorische Metadaten des Patterns Kategorie des Musters, z. B. “HCI” Pattern-Sammlung, die das Muster enthält Klassifizierung des Musters, z. B. nach GoF oder POSA. Das Element wird zunächst als Platzhalter für eine spätere, standardisierte Taxonomie angesehen.
<resulting-context>	Beschreibung des Kontextes, in dem man sich nach Anwendung des Patterns befindet. Dieser kann wiederum ein oder mehrere neue zu lösende Probleme einschließen.

[157] Gang of Four (Viererbande): Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides
[158] Pattern-Oriented Software Architecture
[159] Engl.: Einheitlicher Quellenanzeiger – Spezifikation des Netzwerkprotokolls und des Ortes einer Ressource (z. B. eines Dokuments) in Computernetzwerken (gemäß RFC 1738)

Element/Attribut	Beschreibung
<management>	Metadaten des Musters
<author>	Name des Autors (wie PLML 1.1)
<change-log>	Enthält eine Abfolge von Änderungen
<change>	Änderung der Pattern-Definition
<author>	Autor(en) der Änderung
<description>	Beschreibung der Änderung
<version>	Version der Änderung (ähnlich der Zählung bei Software-Versionen)
<date>	Datum der Änderung
<majorNo>	Nach jeder erheblichen Änderung um den Wert „1“ erhöht
<minorNo>	Nach jeder kleineren Änderung um den Wert „1“ erhöht
<copyright>	Information zum Urheberrecht (wie durch OASIS[160] spezifiziert[161])
<creation-date>	Datum der Erfassung (wie PLML 1.1)
<credits>	Verdienste (wie PLML 1.1)
<last-modified>	Wurde gestrichen, weil nun im Unterelement <version> enthalten
<license>	Lizenzinformation
<license-type>	Lizenz-Typ, z. B. “GNU Free Documentation License”
<ulink>	URL der Lizenzvereinbarung
<revision-number>	Versionsnummer der Pattern-Definition (wie PLML 1.1)

3.2.1.4. XPLML

Ein anderer Ansatz zur Erweiterung von PLML stellt die ebenfalls *Extended Pattern Language Markup Language* genannte Pattern-Beschreibungssprache XPLML dar. Erklärtes Ziel der Sprache ist, sowohl Autoren von Mustern als auch Software-Entwickler effizient beim Umgang mit Patterns zu unterstützen. Neben der Bereitstellung eines einheitlichen Beschreibungsformats für Patterns konzentriert sich XPLML auf die Definition und Veranschaulichung der Beziehungen zwischen den Mustern. Hierzu bedient sich XPLML offener Standards wie XML, *Resource Description Framework*[162] (RDF) und *Web Ontology Language*[163] (OWL), wobei die beiden Letzteren Empfehlungen des *World Wide Web Consortium*[164] (W3C) sind [176]. Mit OWL können Vokabulare definiert werden, durch die wiederum die Ressourcen beschrieben werden können. RDF ist eine einfache Sprache, um Beziehungen zwischen jeweils zwei mittels *Uniform Ressource Identifier*[165] (URI) bestimmten Ressourcen auszudrücken [177].

Das geplante XPLML-Framework umfasst alle Mechanismen und Spezifikationen, die zur Formalisierung von HCI-Patterns benötigt werden. Es wird als das fehlende Glied zwischen den in Prosaform beschriebenen Mustern und der eigentlichen Softwareanwendung gesehen. Folgende sieben Themenblöcke wurden identifiziert, damit XPLML definiert werden kann [176]:

[160] Organization for the Advancement of Structured Information Standards
[161] Siehe http://www.oasis-open.org/docbook/xml/simple/sdocbook/elements/copyright.html
[162] Siehe http://www.w3.org/RDF/
[163] Siehe http://www.w3.org/OWL/
[164] Siehe http://www.w3.org/
[165] Engl.: Einheitliche Bezeichner für Ressourcen gemäß RFC 3986

1. Einheitliches Beschreibungsformat für Patterns
   Definition eines generischen Beschreibungsformats auf Basis von PLML und den Ergebnissen einer noch durchzuführenden Analyse bestehender Pattern-Sammlungen.
2. Semantische Metadaten
   Anreichern der Patterns um semantische Metadaten, um die Suche und das Auffinden der besten Lösung zu einem gegebenen Problem zu erleichtern und zu beschleunigen.
3. Semantische Beziehungen zwischen Patterns
   Formale und maschinell zu verarbeitende Repräsentation unterschiedlicher Arten von Beziehungen.
4. Atomare Teilchen von HCI-Mustern
   Analog zu den durch Christopher Alexander definierten fünfzehn atomaren geometrischen Elementen, durch deren Kombination jedes beliebige Gebäude entworfen werden kann [178], wird bei XPLML davon ausgegangen, dass auch jegliches HCI-Pattern durch eine endliche Menge von Grundelementen zusammengesetzt werden kann.
5. Anforderungsanalyse bei Benutzern von HCI-Mustern
   Erforschung der aktuellen Verwendung von HCI-Patterns in der Industrie und anschließende Optimierung von XPLML.
6. Studie über Management-Werkzeuge für HCI-Muster
   Erforschung der Funktionsweisen von Management-Werkzeugen für Patterns und anschließende Verbesserung der Unterstützung dieser Werkzeuge durch XPLML.
7. Dokumentation der Spezifikation
   Erstellung und Veröffentlichung einer ausführlichen XPLML-Dokumentation.

3.2.1.5. PCML

Auch die von der Firma *ObjectVenture Inc.* entwickelte Sprache *Pattern and Component Markup Language* (PCML) definiert ein standardisiertes Format zur Beschreibung von Patterns jeglichen Abstraktionsgrades [179].

Ähnlich wie bei PLML 1.1 sind auch bei PCML Beschreibungselemente bzw. –Attribute für die Festlegung von Pattern-Name (<name>), alternative Namen (<akas>), Problemstellung (<problem>), Kontext (<context>), Motivation bzw. Randbedingungen (<forces>), Lösung (<solution>) und Beziehungen zu anderen Mustern (<relationships>) vorhanden. Für die Angaben zu Autoren und Version des Patterns, die bei PLML 1.1 in Unterelementen von <management> gemacht werden, stehen die separaten Beschreibungselemente <authors> und <version> zur Verfügung [179]. Während PLML bei der Spezifikation der meisten der genannten Elemente einfachen Prosatext vorsieht, erfolgt die Definition bei PCML mithilfe von Unterelementen deutlich strukturierter.

Zusätzlich zu den Möglichkeiten in PLML 1.1 steht bei PCML zur Identifikation des Patterns das Attribut <namespace> zur Verfügung, durch das ein Namensraum spezifiziert werden kann. Zwei weitere Attribute <abstraction> und <domain> legen Abstraktionsgrad des Musters und die Anwendungsdomäne fest. Das Beschreibungselement <keywords> lässt eine gewisse Klassifizierung des Patterns zu und unterstützt das spätere Suchen und Wiederauffinden des Musters. Das Element <consequences> erlaubt die Angabe von positiven oder auch negativen Folgen, die die Anwendung des Patterns mit sich bringt. Mit dem

BeschreibungeElement <artifacts> können schließlich noch externe Quellen mit weiteren, nicht zwingend im XML-Format vorliegenden, Informationen und Beschreibungen des Musters referenziert werden [179]. Die Elemente bzw. Attribute zur Beschreibung von Patterns sind in Tabelle 8 aufgelistet und erklärt.

Tabelle 8 PCML BESCHREIBUNGSELEMENTE UND -ATTRIBUTE

Element/Attribut	Beschreibung
<namespace>	Namensraum innerhalb dessen der Pattern-Name eindeutig sein muss
<name>	Name des Musters
<abstraction>	Abstraktionsgrad, wie z. B. „Architectural“ oder „Design“
<domain>	Domäne, in der das Muster hauptsächlich eingesetzt werden kann, z. B. „Financial“, „Telecommunications“ oder „Medical“

<authors>
  <author>
  <name>
  <organization>
  <description>
  <URL>
  <display-name>
  <address>
	Autoren des Musters
	  Angabe zu einem Autor
	  Name des Autors
	  Organisation, die durch den Autoren repräsentiert wird
	  Beschreibung des Autors
	  URL zu weiterführenden Informationen oder Email-Adresse
	  Leserfreundlicher Name der URL
	  Die eigentliche URL

<version>
  <revision>
  <date>
  <description>
  <release-notes>
  <copyright>
  <license>
  <artifacts>
	Information über die Version des Musters
	  Versionsnummer
	  Datum und Zeit der Revidierung
	  Beschreibung der Revidierung
	  Beschreibung wichtiger Aspekte der Revidierung
	  Urheberrechtshinweis
	  Lizenzinformation
	  Weitergehende, externe Beschreibung der Revidierung (Details siehe Element <artifacts>)

<akas> <aka>	Weitere Namen für das Muster Weiterer Name, unter dem das Muster bekannt ist
<keywords> <keyword>	Kategorisierung oder Klassifizierung des Musters Schlagwort zur Charakterisierung des Musters

<context>
  <summary>
  <description>
	Beschreibung der Umgebung des Musters
	  Überschrift oder Zusammenfassung der Beschreibung des Musters
	  Ausführliche Beschreibung des Kontexts

<forces>
  <force>
  <summary>
 	
  <description>
	Motivation für das Muster
	  Randbedingung für die Lösung des gegebenen Problems
	  Überschrift oder Zusammenfassung der Beschreibung der Randbedingung
	  Ausführliche Beschreibung der Randbedingung

<problem>
  <summary>
  <description>
	Beschreibung des vom Pattern zu lösenden Problems
	  Kurzer Überblick über die Problemsituation
	  Ausführliche Beschreibung des Problems

Element/Attribut	Beschreibung
<solution>	Lösung des im Pattern aufgezeigten Problems
<summary>	Kurzübersicht über die Lösung
<description>	Ausführliche Beschreibung der Lösung
<participants>	Teilnehmer oder Rollen in der Lösung
<name>	Eindeutiger Name des Teilnehmers
<required>	Hinweis, ob der Teilnehmer für die Lösung unbedingt erforderlich ist
<description>	Beschreibung des Teilnehmers und seiner Rolle bei der Lösung
<stucture>	Statische Struktur der Lösung
<description>	Beschreibung der statischen Interaktion zwischen Teilnehmern
<artifacts>	Weitergehende, externe Beschreibung, z. B. ein UML Klassendiagramm (Details siehe Element <artifacts>)
<collaboration>	Dynamische Aspekte der Lösung
<description>	Beschreibung der dynamischen Interaktion zwischen Teilnehmern
<artifacts>	Weitergehende, externe Beschreibung, z. B. ein UML Sequenzdiagramm (Details siehe Element <artifacts>)
<consequences>	Konsequenz des Einsatzes des Musters
<consequence>	Jeweils ein Pro oder Contra für den Einsatz des Musters
<summary>	Überschrift oder Zusammenfassung der Beschreibung der Konsequenz
<description>	Ausführliche Beschreibung der Konsequenz
<relationships>	Weitere Muster, zu denen eine Beziehung besteht
<relationship>	Beziehung zwischen zwei Mustern
<namespace>	Namensraum des in Beziehung stehenden Musters
<name>	Name des in Beziehung stehenden Musters
<type>	Typ der Beziehung
<summary>	Kurzbeschreibung des in Beziehung stehenden Musters
<description>	Beschreibung, wie die beiden Muster in Beziehung zueinander stehen
<artifacts>	Externe Quellen mit weiteren Beschreibungen des Musters
<artifact>	Externe Datei, die nicht zwingend in XML-Format vorliegt
<name>	Name des Artefakts
<type>	Dateityp des Artefakts bezeichnet durch eine gebräuchliche Dateiendung, z. B. „doc“ oder „html“
<description>	Beschreibung des Artefakts

Keine direkten Entsprechungen in PCML gibt es für die PLML 1.1-Elemente <synopsis> und <confidence>. Das ebenfalls fehlende PLML 1.1-Attribut <patternID> wird nicht benötigt, da ein Muster in PCML durch die Kombination von <namespace> und <name> eindeutig identifizierbar ist. Die Informationen über die Anwendung und Implementierung des Patterns, die bei PLML 1.1 in den Elementen bzw. Unterelementen <example>, <rationale>, <illustration>, <implementation> und gegebenenfalls auch <diagram> enthalten sind, werden bei PCML nicht direkt in der Pattern-Definition, sondern in einem separaten Konstrukt, einer sogenannten Strategie, beschrieben.

Ein Pattern kann über mehr als eine Strategie verfügen, von der jede eine bestimmte Implementierung der durch das Muster vorgegebenen Problemlösung repräsentiert. Durch die Auslagerung der Implementierungsdetails aus der eigentlichen Pattern-Beschreibung gibt es keine Einschränkungen bezüglich der Anzahl von Strategien zu einem Muster. Zudem

können zu einem späteren Zeitpunkt auch andere Personen als der Pattern-Autor ihre Strategien beschreiben, ohne dass die Pattern-Spezifikation angepasst werden muss. Eine Strategie wird spezifiziert mittels Informationen über den Namensraum (<namespace>), Namen (<name>), alternative Namen (<akas>), Beschreibung (<description>), Autoren (<authors>), Versionsabgaben (<version>), Kategorisierung bzw. Klassifizierung (<keywords>), Rollen (<roles>), das Muster, das durch die Strategie implementiert wird (<pattern-ref>), andere Strategien, aus der sich die vorliegende zusammensetzt (<strategies>) und externe Quellen mit weitergehenden Beschreibungen der Strategie (<artifacts>) [179].

3.2.1.6. TPML

Die *Task Pattern Markup Language* (TPML) wurde an der *Concordia University* in Montreal entwickelt und dient zur Spezifikation von Aufgaben-Mustern [180]. Praktische Anwendung findet TPML im Kontext des Pattern-basierten Modellierungs-Werkzeugs *PIM Tool* (siehe Kapitel 3.4.2.7), das in diesem Beschreibungsformat vorliegende Muster verarbeiten kann [20].

Die Beschreibung der Muster erfolgt grundsätzlich mittels einer Untermenge der in PLML 1.1 definierten Beschreibungselemente <Name>, <Problem>, <Context>, <Solution> und <Rationale>. Diese dienen bei TPML hauptsächlich dem Auffinden und der Selektion der für den jeweiligen Anwendungsfall geeigneten Patterns. Darüber hinaus beherbergt das Element <Body>, das bei PLML 1.1 keine Entsprechung findet, die eigentlich zur Modellierung benötigte Formalisierung des Musters [180]. Die zum Einsatz kommenden Beschreibungselemente sind zusammen mit der jeweiligen Erklärungen in Tabelle 9 aufgelistet.

Tabelle 9 TPML BESCHREIBUNGSELEMENTE UND -ATTRIBUTE

Element/Attribut	Beschreibung
<Name>	Name des Musters
<Problem>	Beschreibung des zu lösenden Problems
<Context>	Situation, in der das Muster angewendet werden kann
<Solution>	Beschreibung, wie das Problem zu lösen ist
<Rationale>	Begründung dafür, warum die Lösung das Problem löst
<Body>	Hauptteil mit der Aufgabenbeschreibung
<Task>	Statische Aufgabenbeschreibung
<ID>	Eindeutiger Bezeichner der Aufgabe
<Name>	Name der Aufgabe
<Type>	Typ der Aufgabe
<Category>	Kategorie der Aufgabe
<Relation>	Zeitliche Beziehung der Aufgabe
<Order>	Anzeigereihenfolge für die Visualisierung der Aufgabe
<SubTasks>	Komposition aus <Task>- und <TaskTemplate>-Elementen
<TaskTemplate>	Variable Aufgabenbeschreibung
<ID>	Eindeutiger Bezeichner der Aufgabe
<VariableDef>	Deklaration neuer Variablen
<Type>	Datentyp der Variablen
<Name>	Name der Variablen
<Description>	Beschreibung der Variablen
<Name>	Name der Aufgabe
<Text>	Textanteil des Namens der Aufgabe

Element/Attribut	Beschreibung
<Variable>	Name der Variablen
<Type>	Typ der Aufgabe
<Category>	Kategorie der Aufgabe
<Relation>	Zeitliche Beziehung der Aufgabe
<Order>	Anzeigereihenfolge für die Visualisierung der Aufgabe
<SubTasks>	Komposition aus <Task>- und <TaskTemplate>-Elementen

Das Beschreibungselement <Body> setzt sich aus den zwei Unterelementen <Task> und <TaskTemplate> zusammen. Das komplexe Element <Task> nimmt hierbei statische Aufgaben-Beschreibungen, die keiner weiteren Anpassung mehr bedürfen, auf und besteht aus einer Hierarchie von Unteraufgaben, die wiederum aus <Task>- und/oder <TaskTemplate>-Elementen aufgebaut sein können. Das ebenfalls komplexe Element <TaskTemplate> ist ähnlich wie <Task> aufgebaut, ist aber dazu gedacht, variable Anteile von Aufgaben-Beschreibungen zu repräsentieren. Hierzu dient das zusätzliche Unter-Element <VariableDef>, das wiederum aus einem Namen (Unter-Element <Name>) und einer Beschreibung (Unter-Element <Description>) besteht und zudem über das Attribut <Type> verfügt, das den Datentyp der betreffenden Variablen festlegt. Zudem ist das Unter-Element <Name>, anders als bei <Task>, zusammengesetzt aus den Unter-Elementen <Text> und <Variable>. Bei letzterem handelt es sich um die diejenigen Variablen, die zuvor im Teil <VariableDef> definiert wurden. Bei der Nutzung des Musters werden dann die variablen Anteile aufgelöst, d.h. es erfolgt eine Anpassung an den vorliegenden Anwendungskontext. Nach diesem Vorgang erhält man schließlich eine adaptierte, statische Aufgaben-Beschreibung, die ihrer weiteren Verwendung zugeführt werden kann [180].

3.2.2. Vergleich und Bewertung

Die Pattern-Beschreibungssprache PLML 1.1 stellt 17 Beschreibungselemente dar, von denen drei durch Unter-Elemente weiter strukturiert sind. Auf dieser Basis können die Muster schon recht umfassend und detailliert dokumentiert werden. Der Bezeichner <patternID> erlaubt die eindeutige Identifikation von Mustern innerhalb einer Pattern-Sammlung, nicht aber über deren Grenzen hinweg, da hierzu notwendige Vorgaben an den Bezeichner fehlen. Zu einer inhaltlichen Überschneidung kommt es bei den Beschreibungselementen <illustration> und <example>, da beide zur Dokumentation von Anwendungsbeispielen des betreffenden Patterns dienen. Ähnliches gilt für die Beschreibungselemente <related-patterns> und <pattern-link>, durch die jeweils Beziehungen zu anderen Mustern definiert werden. Sie unterscheiden sich lediglich durch den Grad der Strukturierung ihrer Inhalte. Ersteres enthält nicht weiter strukturierte Beschreibungen, während Zweiteres durch die Unter-Elemente <type>, <patternID>, <collection> und <label> deutlichere Vorgaben hinsichtlich der erwarteten Informationen macht. Anhand des Elements <confidence> soll die Einschätzung der Wahrscheinlichkeit, dass die angegebene Lösung das durch das Pattern adressierte Problem tatsächlich löst, angegeben werden. Eine Erklärung, wie solch ein Wert zustande kommt, ist aber nicht vorhanden. Eine Idee hierzu wäre, Ergebnisse aus Benutzerbefragungen und/oder Usability-Tests als Basis für eine derartige Aussage heranzuziehen. Bezüglich PLML 1.1 fällt auf, dass neben den bereits erwähnten Beschreibungselementen <example> und <related-patterns> auch <alias>, <forces>, <diagram>, <implementation>, <author> und <literature> nicht

weiter strukturiert sind. Zum einen ist dies von Nachteil, weil zu erwarten ist, dass diesen Elementen mehrere Einträge zugeordnet werden sollen und somit jeweils eine Liste von Begriffen vorhanden ist. Dies würde insbesondere bei einer automatisierten Verarbeitung der Pattern-Beschreibungen zu vermeidbaren Aufwänden, z. B. für Parsing, führen. Zum anderen würde eine vorgegebene Strukturierung eine Vergleichbarkeit von Pattern-Spezifikationen verschiedener Autoren fördern.

Hinsichtlich der oben genannten Kritikpunkte birgt PLML 1.2 einige Verbesserungen. Die Beschreibungselemente <forces>, <example>, <implementation> und <literature> besitzen nun eine vorgegebene Struktur und geben somit eine bessere Orientierung bezüglich der angedachten Inhalte. Zudem sind dadurch die jeweiligen Einträge von etwaigen Auflistungen einzeln zugreifbar. Das zusätzliche Unter-Element <revision-number> des Beschreibungselements <pattern-link> erlaubt nun die Herstellung genauerer Bezüge zwischen den Mustern. Darüber hinaus verfügt das Element <management> nun über ein strukturiertes Change-Log, das die Änderungen an der Pattern-Spezifikation chronologisch nachvollziehbar macht. Die Beschreibungselemente <alias>, <related-patterns>, <diagram> und <author> bleiben aber nach wie vor unstrukturiert. Das neu eingeführte Element <collection> erlaubt die Benennung der Pattern-Sammlung, zu der das vorliegende Muster gehört.

Im Rahmen von PLMLx entfällt das PLML 1.1-Element <evidence> und dessen beide Unter-Elemente <example> und <rationale> werden nun als Beschreibungselemente erster Ebene geführt. Durch das neu hinzugekommene, strukturierte Element <organization> werden die Möglichkeiten hinsichtlich der Spezifikation von Metadaten verbessert. Das ebenfalls neue Element <resulting-context> erlaubt die Beschreibung des nach der Anwendung des betreffenden Patterns bestehenden Kontextes. Ähnlich wie bei PLML 1.2 wird auch bei PLMLx ein strukturiertes Change-Log innerhalb des Beschreibungselements <management> spezifiziert, jedoch auf Basis anderer Unter-Elemente. Zudem erhält <management> mit <copyright> und <license> Unter-Elemente für die Aufnahme von Copyright- und Lizenz-Informationen, das unter-Element <author> bleibt jedoch auch hier weiterhin unstrukturiert. Das zusätzliche Element erster Ebene <acknowledgements> bietet die Möglichkeit, maßgebliche Beteiligungen an der Pattern-Spezifikation zu honorieren.

Die Beschreibungssprache XPLML konzentriert sich auf die Definition und Veranschaulichung der Beziehungen zwischen Patterns. In der hierzu verfügbaren Literatur [176] [177] ist der konkrete Aufbau von XPLML allerdings nicht enthalten, sodass eine diesbezügliche Bewertung an dieser Stelle nicht erfolgen kann.

Bei PCML wird die eindeutige Identifikation von Mustern durch das Beschreibungselement <namespace> unterstützt. Hierdurch wird eine ähnliche Wirkung wie durch das PLML 1.2-Element <collection> erzielt. Die zusätzlichen Elemente <abstraction>, <domain> und <keywords> tragen zur inhaltlichen Ergänzung der Metadaten zu den Patterns bei. Die Beschreibungselemente <context>, <forces>, <problem>, <solution> sind grundsätzlich so strukturiert, dass sowohl eine Orientierung bezüglich der geforderten Inhalte gegeben ist, als auch bei Auflistungen auf die jeweiligen Einträge einzeln zugegriffen werden kann. Letzteres gilt auch für das Element <akas>, das inhaltlich dem PLML 1.1-Element <alias> entspricht. Die Beziehungen zu anderen Patterns werden anhand des ebenfalls strukturierten Elements <relationships> spezifiziert. Hierbei findet u.a. auch der durch <namespace> definierte Namensraum Berücksichtigung. Das neu eingeführte, strukturierte Beschreibungselement <consequences> dient zur Dokumentation von positiven und negativen Aspekten, die durch die Verwendung des Musters entstehen. Das Element

<authors> ist bei PCML in strukturierter Form vorhanden. Ebenfalls in Unter-Elemente gegliedert ist das Beschreibungselement <version>, das auch Freigabe-Vermerke sowie Copyright- und Lizenz-Informationen beinhaltet. Das zusätzliche, strukturierte Element <artifacts> beherbergt Verweise auf weitergehende, externe Beschreibungen und entspricht somit inhaltlich in etwa dem PLML 1.1-Element <literature>.

Die Beschreibungssprache TPML ist spezialisiert auf die Definition von Task-Patterns. Dennoch können aber grundsätzlich alle Arten von Patterns dokumentiert werden, wenngleich hierbei nur vergleichsweise wenige Beschreibungselemente zur Verfügung stehen. Die Besonderheit von TPML zeigt sich aber im Element <Body>, das es erlaubt, statische und variable Anteile von Benutzer-Aufgaben sowie deren temporalen Beziehungen zueinander zu spezifizieren.

3.3. Pattern-Sammlungen

Das folgende Kapitel beschäftigt sich mit bereits existierenden Pattern-Sammlungen. Im folgenden Unterkapitel werden zunächst grundlegende Begrifflichkeiten hierzu definiert. In Kapitel 3.3.2 sind dann die im Rahmen dieser Dissertation im Detail untersuchten Pattern-Sammlungen *Designing Interfaces*, *Patterns in Interaction Design*, *Design of Sites*, *Quince Pattern Library* und *Yahoo Design Pattern Library* beschrieben. Abschließend befindet sich in Kapitel 3.3.3 ein Vergleich und die Bewertung dieser Muster-Sammlungen, insbesondere hinsichtlich der Transformierbarkeit der jeweiligen Pattern-Beschreibungen in das durch PLML 1.1 (siehe Kapitel 3.2.1.1) vorgegebene Format. Die betreffenden Inhalte wurden bereits teilweise in [172] veröffentlicht.

3.3.1. Begriffsdefinitionen

3.3.1.1. Pattern-Katalog

Ein Pattern-Katalog (engl. Pattern Catalogue) umfasst eine Reihe von Mustern, die konsistent und in einer einheitlichen Form beschrieben sind. Zu Zwecken der Übersichtlichkeit und besseren Auffindbarkeit erfolgt eine Kategorisierung. Beziehungen zwischen den Patterns sind aber nicht oder nicht durchgängig definiert. Es können sogar Muster enthalten sein, die in keinerlei Beziehung zu anderen Patterns desselben Katalogs stehen. Pattern-Kataloge decken üblicherweise eine Anwendungs-Domäne nicht vollständig ab. Die Bezeichnungen Muster- bzw. Pattern-Katalog (engl. Pattern Catalogue) und Muster- bzw. Pattern-Kollektion (engl. Pattern Collection) werden synonym verwendet [145].

3.3.1.2. Pattern-Sprache

Im Gegensatz zu einem Pattern-Katalog decken die Muster in einer Muster- bzw. Pattern-Sprache (engl. Pattern Language) eine Familie von Design-Problemen in einer gegebenen Anwendungs-Domäne vollständig ab. Eine Pattern-Sprache beschreibt die domänenspezifischen Probleme anhand von abstrakten Entwurfsmustern (engl. High-level Design Patterns), die wiederum durch konkretere Entwurfsmuster (engl. Low-level Design Patterns) gelöst

werden. Die zwischen den Patterns bestehenden Beziehungen bilden hierbei nicht eine einfache Hierarchie sondern ein ganzes Netzwerk von Relationen [145].

In [143] werden drei wesentliche Eigenschaften von Pattern-Sprachen genannt:

1. Die Beschreibung der Muster erfolgt durchgängig in einer einheitlichen, standardisierten Form.
2. Es erfolgt eine logische Gruppierung der Muster.
3. Die Beziehungen zwischen den Mustern werden beschrieben.

3.3.1.3. Pattern-Sammlung

In der vorliegenden Dissertation wird die Bezeichnung Pattern-Sammlung (engl. Pattern Compilation) als Oberbegriff für Pattern-Kataloge und Pattern-Sprachen definiert, um einen wertfreien Terminus verwenden zu können, wenn eine Unterscheidung zwischen Muster-Katalog und -Sprache nicht erforderlich ist oder aufgrund fehlender Detail-Informationen nicht getroffen werden kann.

3.3.1.4. Pattern-Bibliothek

Manche Muster-Sammlungen tragen in ihren Namen die Bezeichnung Muster- bzw. Pattern-Bibliothek (engl. Pattern Library), z. B. *Quince Pattern Library* (siehe Kapitel 3.3.2.5) oder *Yahoo Design Pattern Library* (siehe Kapitel 3.3.2.4). Dieser Begriff wird in der vorliegenden Dissertation synonym zur Bezeichnung Pattern-Sammlung verwendet, da a priori nicht feststeht, ob es sich um einen Pattern-Katalog oder eine Pattern-Sprache handelt.

3.3.2. Untersuchte Pattern-Sammlungen

3.3.2.1. Designing Interfaces

Die zweite Auflage der Pattern-Sammlung *Designing Interfaces* von Jenifer Tidwell aus dem Jahr 2011 ist in elf Kategorien eingeteilt und umfasst insgesamt 125 Muster. Bei der Überarbeitung kamen einige neue Patterns und drei Kategorien hinzu und viele der bestehenden Patterns wurden revidiert [181]. Ein Auszug ist im Internet verfügbar [182].

Eine Übersicht in Tabellenform zur Organisation der Patterns befindet sich in Anhang C.1. Die dort zusammengefassten Informationen beziehen sich auf die ausführliche Version der Pattern-Sammlung [181]. Die Tabelle beinhaltet die vorhandenen Kategorien, die Anzahl der in der jeweiligen Kategorie enthaltenen Muster sowie die Originalbezeichnungen der Patterns in englischer Sprache.

Die Spezifikation der Muster wird durch die acht Beschreibungselemente <name>, <figure>, <what>, <use-when>, <why>, <how>, <examples> und <in-other-libraries> spezifiziert. Diese Elemente werden in Tabelle 10 aufgelistet und erklärt.

Tabelle 10 BESCHREIBUNGSELEMENTE DER PATTERNS VON JENIFER TIDWELL

Element	Beschreibung
<name>	Name des Musters
<figure>	Aussagekräftiges Beispiel der Anwendung des Musters, beispielsweise in Form eines Screenshots
<what>	Kurze Beschreibung des Musters
<use-when>	Information über das zu lösende Problem und den Kontext der Anwendung des Musters
<why>	Gründe für die Anwendung des vorliegenden Musters
<how>	Beschreibung, wie das vorliegende Muster angewendet werden soll
<examples>	Beschreibung von Anwendungsbeispielen
<in-other-libraries>	Verweise auf andere Pattern-Sammlungen, die dasselbe Muster enthalten

3.3.2.2. Patterns in Interaction Design

Die Pattern-Sammlung *Patterns in Interaction Design* von Martijn van Welie ist im Internet verfügbar [183]. Sie besteht aus insgesamt 131 Mustern, die in drei Haupt- und insgesamt 15 Unterkategorien organisiert sind.

In Anhang C.2 befindet sich eine Übersicht zur Organisation der Patterns anhand von drei Tabellen. In der ersten Tabelle werden für die Hauptkategorie „User needs“ die enthaltenen Unterkategorien, die Anzahl der in der jeweiligen Unterkategorie verfügbaren Muster sowie die Originalbezeichnungen der Patterns in englischer Sprache angegeben. Analog hierzu zeigen die beiden weiteren Tabellen diese Informationen jeweils für die Hauptkategorien „Application needs“ und „Context of design“ [183].

Die Muster werden durch die neun Elemente <name>, <problem>, <solution>, <use-when>, <how>, <why>, <more-examples>, <literature> und <comments> beschrieben. Details zu diesen Beschreibungselementen befinden sich in Tabelle 11.

Tabelle 11 BESCHREIBUNGSELEMENTE DER PATTERNS VON MARTIJN VAN WELIE

Element	Beschreibung
<name>	Name des Musters
<problem>	Beschreibung des zu lösenden Problems
<solution>	Beschreibung der vorgeschlagenen Lösung zum gegebenen Problem
<use-when>	Information über den Kontext der Anwendung des Musters
<how>	Beschreibung, wie das vorliegende Muster angewendet werden soll
<why>	Gründe für die Anwendung des vorliegenden Musters
<more-examples>	Beschreibung von Anwendungsbeispielen des vorliegenden Musters
<literature>	Verweise auf Literatur mit Hintergrundinformation
<comments>	Kommentare von Personen, die das Muster verwendet haben

3.3.2.3. Design of Sites

Die Pattern-Sammlung *Design of Sites* von Douglas van Duyne besteht aus 107 Mustern, die in insgesamt 13 Kategorien eingeteilt sind [147]. Alle Patterns sind in verkürzter Form auch im Internet veröffentlicht [184].

Eine Übersicht in Tabellenform zur Organisation der Patterns befindet sich in Anhang C.3. Die dort angegebenen Informationen beziehen sich wiederum auf die ausführliche Version der Pattern-Sammlung [147]. Die Tabelle enthält die vorhandenen Kategorien, die Anzahl der in der jeweiligen Kategorie enthaltenen Muster sowie die Originalbezeichnungen der Patterns in englischer Sprache.

Die Muster werden mithilfe der sieben Beschreibungselemente <patternID>, <name>, <figure>, <background>, <problem>, <solution> und <other-patterns-to-consider> spezifiziert. In Tabelle 12 befinden sich die jeweiligen Beschreibungen hierzu.

Tabelle 12 BESCHREIBUNGSELEMENTE DER PATTERNS VON DOUGLAS VAN DUYNE

Element	Beschreibung
<patternID>	Eindeutiger Bezeichner des Musters
<name>	Name des Musters
<figure>	Aussagekräftiges Beispiel der Anwendung des Musters, beispielsweise in Form eines Screenshots
<background>	Information über den Kontext der Anwendung des Musters
<problem>	Beschreibung des zu lösenden Problems
<solution>	Beschreibung der vorgeschlagenen Lösung zum gegebenen Problem
<other-patterns-to-consider>	Liste von Patterns, die mit dem vorliegenden Muster in Beziehung stehen

Die Online-Version der Pattern-Sammlung verfügt über das zusätzliche Beschreibungselement <comments>, in dem Kommentare von Anwendern des Patterns hinterlegt werden können [184].

3.3.2.4. Yahoo Design Pattern Library

Die *Yahoo Design Patterns Library* der Firma *Yahoo! Inc.* ist im Internet verfügbar [185]. Sie besteht aus insgesamt 55 Mustern, die in fünf Haupt-, neun Unter- und sieben Unter-Unterkategorien organisiert sind.

In Anhang C.5 befinden sich drei Tabellen, die einen Überblick über die Organisation der Patterns geben. Die erste Tabelle zeigt die Hauptkategorien, deren Unterkategorien, die Anzahl der in der jeweiligen Kategorie bzw. Unterkategorie enthaltenen Muster sowie die Originalbezeichnungen der Patterns in englischer Sprache. Analog zeigen die beiden folgenden Tabellen diese Informationen jeweils für die Unter-Unterkategorien der Unterkategorien „People“ und „Objects“ [185].

Die Patterns werden mittels der 15 Beschreibungselemente <name>, <status>, <last-modified>, <illustration>, <problem> (What Problem Does This Solve?”), <context> (When to Use This Pattern), <solution> (What’s the Solution?), <why> (Why to Use This Pattern?),

<special-cases, <examples>, <pattern-gallery>, <accessibility>, <disambiguation>, <open-questions> und <pattern-information> spezifiziert. Bei den jeweils in runden Klammern genannten Angaben handelt es sich um die original verwendeten Bezeichnungen. Das Element <pattern-information> enthält zusätzliche Informationen zum Pattern und stellt hierfür sieben Unterelemente zur Verfügung. Die Beschreibungselemente werden in Tabelle 13 aufgelistet und erklärt.

Tabelle 13 BESCHREIBUNGSELEMENTE DER PATTERNS DER YAHOO DESIGN PATTERN LIBRARY

Element	Beschreibung
<name>	Name des Musters
<status>	Statusinformation zum Pattern, z. B. „Best Practice“, „Working Solution“ oder „Beta“
<last-modified>	Datum der letzten Änderung
<illustration>	Aussagekräftiges Beispiel der Anwendung des Musters in textueller und grafischer Form
<problem>	Beschreibung des zu lösenden Problems
<context>	Information über den Kontext der Anwendung des Musters
<solution>	Beschreibung der vorgeschlagenen Lösung zum gegebenen Problem
<why>	Gründe für die Anwendung des vorliegenden Musters
<special-cases>	Beschreibung von bei der Anwendung des Musters zu beachtende Details
<examples>	Beschreibung von weiteren Anwendungsbeispielen des vorliegenden Musters
<pattern-gallery>	Beispiele in Form von Bildern
<accessibility>	Beschreibung von durch den Benutzer wahrnehmbare Implementierungsdetails
<disambiguation>	Begriffserklärung zur Vermeidung von Doppeldeutigkeiten bzw. Verwechslung von Patterns
<open-questions>	Liste von für die Pattern-Definition noch zu klärenden Punkten (Zulässigkeit dieses Elements in Abhängigkeit von <status>)

<pattern-information>
  <as-used-on-Yahoo>
  <other-examples>
  <code-examples>
  <related-patterns>
  <similar-patterns-in-other-libraries>
  <wire-frame-stencils>
  <blog>
	Zusätzliche Informationen zu den Mustern
	  Beispiele aus Yahoo-Produkten
	  Beispiele aus anderen Anwendungen oder Produkten
	  Anwendungsbeispiele in Form von Code-Fragmenten
	  Beziehungen zu anderen Mustern in der Yahoo Design Pattern Library
	  Beziehungen zu Mustern in anderen Pattern-Sammlungen
	  Drahtmodell-Schablonen, die im Entwurfsprozess (z. B. für Skizzen) verwendet werden können
	  Verweise auf Weblogs mit Kommentaren zum Pattern

3.3.2.5. Quince Pattern Library

Die *Quince User Experience* (UX) *and User Interface* (UI) *Pattern Library* der Firma *Infragistics Inc.* umfasst 92 Muster, die durch insgesamt 43 verschiedene Tags gekennzeichnet und

somit kategorisiert sind. In der Regel besitzen die Muster mehrere Tags, sodass sie mehr als einer Kategorie zugeordnet werden. Die Pattern-Sammlung ist im Internet verfügbar [186].

Eine Übersicht über die in der Muster-Sammlung enthaltenen Patterns, deren Originalbezeichnungen in englischer Sprache sowie deren Zuordnung zu den definierten Tags bzw. Kategorien befindet sich in Anhang C.4 [186].

Die Muster werden mithilfe der zehn Beschreibungselemente <name>, <problem>, <solution>, <context>, <rationale>, <implementation>, <examples>, <sources>, <comments> und <tags> spezifiziert. In Tabelle 14 befinden sich die jeweiligen Beschreibungen hierzu.

Tabelle 14 BESCHREIBUNGSELEMENTE DER QUINCE PATTERN LIBRARY

Element	Beschreibung
<name>	Name des Musters
<problem>	Beschreibung des zu lösenden Problems
<solution>	Beschreibung der vorgeschlagenen Lösung zum gegebenen Problem
<context>	Information über den Kontext der Anwendung des Musters
<rationale>	Begründung dafür, dass die beschriebene Lösung das gegebene Problem im gegebenen Kontext tatsächlich löst
<implementation>	Hinweise und Details hinsichtlich der praktischen Realisierung in Textform
<examples>	Beschreibung von Anwendungsbeispielen des vorliegenden Musters
<sources>	Verweise auf Quellen, die zur Beschreibung des Musters herangezogen wurden, meist Referenzen auf andere Pattern-Sammlungen
<comments>	Kommentare von Personen, die das Muster verwendet haben
<tags>	Kennzeichen zur Kategorisierung der Muster

3.3.3. Vergleich und Bewertung

Die Pattern-Sammlung *Designing Interfaces* von Jenifer Tidwell kann für eine Vielzahl von Anwendungs-Domänen eingesetzt werden. Die Muster werden im Detail erklärt und sind gut illustriert. Zu allen Patterns sind aussagekräftige Anwendungsbeispiele vorhanden. Sprechende Bezeichnungen für die Kategorien erleichtern die Suche nach bestimmten Mustern. Informationen über Beziehungen zu anderen Patterns werden nicht durchgängig angegeben, sind aber mitunter in den Beschreibungselementen <how> and <why> vorhanden. Referenzen auf dasselbe oder sehr ähnliche Muster in anderen Sammlungen werden mithilfe des Elements <in-other-libraries> spezifiziert. Aufgrund des fehlenden Bezugs zu einer bestimmten Anwendungs-Domäne und der nur sporadisch definierten Pattern-Beziehungen ist *Designing Interfaces* als Pattern-Katalog einzuordnen.

Im Gegensatz zu der Pattern-Sammlung von Jenifer Tidwell wurde *Patterns in Interaction Design* von Martijn van Welie schon seit einigen Jahren nicht mehr gepflegt. Laut der Webseite [183] fand die letzte Überarbeitung im Jahr 2008 statt. Die zweistufige Kategorisierung erleichtert die Suche und das Auffinden der Muster. Auch hier gibt es kein ausgewiesenes Beschreibungselement für die Spezifikation von Beziehungen zwischen Patterns. Zwar sind Informationen zu Relationen zwischen Mustern innerhalb der Sammlung mitunter in den

Elementen <use-when>, <how> oder <why> zu finden, Referenzen zu anderen Pattern-Katalogen fehlen aber gänzlich. Ein nützliches Instrument ist das Element <comments>, das Anmerkungen von Anwendern der Muster enthält, die über die Webseite eingefügt werden können. Auch *Patterns in Interaction Design* ist wegen des fehlenden Bezugs zu einer bestimmten Anwendungs-Domäne und der nur sporadisch spezifizierten Relationen zwischen den Mustern als Pattern-Katalog zu bezeichnen.

Auch die Muster-Sammlung *Design of Sites* von Douglas van Duyne wurde schon längere Zeit nicht mehr aktualisiert. Die letzte Ausgabe des Buches [147] stammt aus dem Jahr 2006 und laut der Webseite [184] fand hier die letzte Überarbeitung im Jahr 2009 statt. Die Zielgruppe dieser Pattern-Sammlung sind Web-Designer [147]. Dies wird auch bereits durch die Namensgebung für die Pattern-Kategorien angedeutet. Die Beschreibungen der Muster sind umfangreich und detailliert. Positiv zu bewerten ist die Darstellung der Abhängigkeiten und Beziehungen zu Mustern innerhalb der Sammlung mithilfe des Beschreibungselements <other-patterns-to-consider>. Jedoch fehlen auch hier Referenzen zu anderen Pattern-Sammlungen. In der Online-Version gibt es analog zu *Patterns in Interaction Design* das Element <comments>, das Kommentare von Nutzern der Patterns aufnimmt. *Design of Sites* umfasst Patterns für die Domäne der Webseiten-Gestaltung, deren Beziehungen zueinander wohldefiniert sind. Es handelt sich deshalb um eine Pattern-Language.

In der *Quince Pattern Library* werden die dort enthaltenen Muster präzise beschrieben und werden sehr übersichtlich präsentiert. Für jedes Pattern wurden ein oder mehrere Tags vergeben, durch die das Auffinden von Mustern und die Identifikation von in Beziehung stehenden Patterns deutlich erleichtert werden. Ähnlich wie bei den Muster-Sammlungen von van Welie und van Duyne steht für registrierte Benutzer das Element <comments> zur Verfügung, mit dessen Hilfe Anmerkungen und Verbesserungsvorschläge dokumentiert werden können. Zudem besteht die Möglichkeit, den Patterns über das Beschreibungselement <examples> praktische Anwendungsbeispiele hinzuzufügen, sodass die meisten vorhandenen Muster diesbezüglich sehr ausführlich dokumentiert sind. Die Aktualität der Pattern-Beschreibungen lässt sich anhand der durch die Webseite [186] bereitgestellten Informationen nicht ermitteln. Der aktuellste Benutzerkommentar wurde am 5. März 2014 hinterlassen. Der überwiegende Anteil der Kommentare ist deutlich älter und stammt aus dem Jahr 2009. Da zwar die Beziehungen zwischen den Mustern wohldefiniert sind, der Pattern-Sammlung aber der Bezug zu einer bestimmten Anwendungs-Domäne fehlt, ist die *Quince Pattern Library als* Pattern-Katalog einzuordnen.

Die Muster in der *Yahoo Design Pattern Library* beinhalten sehr ausführliche Beschreibungen der Lösungsansätze, die meist auch Informationen hinsichtlich möglicher Implementierungen enthalten. Weitergehende Tipps zur Realisierung werden zudem in einer Reihe anderer Beschreibungselemente bereitgestellt. Die Patterns sind durch viele Beispiele illustriert. Mithilfe des Elements <blogs> können bei den meisten Muster Kommentare in Weblogs eingesehen werden. Nützlich sind die Beschreibungselemente <status> und <open-questions>. Ersteres gibt Auskunft über den Reifegrad der betreffenden Pattern-Spezifikation; falls sich die Beschreibung noch in einem Entwurfsstatus befindet, so können in letzterem die noch zu klärenden Punkte hinterlegt werden. Im sporadisch verfügbaren Element <wire-frame-stencils> befinden sich Verweise auf Drahtmodell-Schablonen, die im Design-Prozess, z. B. zur Erstellung von Skizzen, verwendet werden können. Auch diese Pattern-Sammlung wird schon jahrelang nicht mehr gepflegt. Die letzten Änderungen fanden gemäß der Angaben im Beschreibung-Element <last-modified> ausnahmslos im

Jahr 2009 statt. Die *Yahoo design Pattern Library* ist als Pattern-Katalog einzuordnen, da Beziehungen zwischen den Mustern zwar in einigen Fällen definiert sind, der Muster-Sammlung aber insgesamt der Bezug zu einer bestimmten Anwendungs-Domäne fehlt.

Die Beschreibungselemente aller fünf untersuchten Pattern-Sammlungen können fast vollständig den Elementen von PLML 1.1 zugeordnet werden. In einigen Fällen handelt es sich um 1:1-Beziehungen, jedoch sind auch mehrdeutige Zusammenhänge zu beobachten, d.h. 1:n-, n:1- oder n:m-Zuordnungen. Ohne Entsprechung in PLML 1.1 sind folgende vier Informationsarten:

1. Kommentare
   Anmerkungen von Anwendern der Patterns (Element <comments> bei *Patterns in Interaction Design*, der Online-Version von *Design of Sites* und der *Quince Pattern Library*) bzw. Verweise auf Weblog-Einträge (Element <blogs> der *Yahoo Design Pattern Library*).
2. Kennzeichnung
   Vergabe von Kennzeichen in Form von Tags zur Bildung von Pattern-Kategorien und der Modellierung von Beziehungen zwischen Mustern (Element <tags> der *Quince Pattern Library*).
3. Reifegrad
   Angabe des Bearbeitungsstatus bzw. des Reifegrades der Pattern-Beschreibung, insbesondere wenn auch Entwürfe von Mustern in der Sammlung enthalten sind (Element <status> der *Yahoo Design Pattern Library*) und Dokumentation der zu klärenden Punkte, die zur Vervollständigung der Beschreibung noch notwendig sind (Element <open-questions> der *Yahoo Design Pattern Library*).
4. Verfügbare Werkzeuge
   Verweise auf für die Anwendung des Musters im Design-Prozess zur Verfügung stehende Werkzeuge, im vorliegenden Fall auf Drahtmodell-Schablonen zur Erstellung von Skizzen (Element <wire-frame-stencils> der *Yahoo Design Pattern Library*).

Andererseits bleiben die vier PLML 1.1-Elemente <alias>, <forces>, <synopsis> und <confidence> bei allen fünf betrachteten Pattern-Sammlungen gänzlich unberücksichtigt. Zudem finden sich hinsichtlich der komplexen PLML 1.1-Elemente <pattern-link> und <management> nur jeweils ein Teilaspekt in *Design of Sites* bzw. in der *Yahoo Design Pattern Library* wieder.

In der Sammlung *Designing Interfaces* von Tidwell beinhaltet das Element <use-when> Informationen, die sich bei PLML 1.1 teilweise <problem> und <context> zuordnen lassen. Andererseits besitzen die Elemente <what> und <how> Inhalte, die auf <solution> in PLML 1.1 abgebildet werden können. Ferner liefert das Element <in-other-libraries> bei Tidwell in einigen Fällen Informationen, die in <literature> bei PLML 1.1 passt. Über die oben genannten Elemente hinaus bleiben <patternID>, <diagram> und <implementation> unberücksichtigt.

Van Welie präsentiert bei *Patterns in Interaction Design* bezüglich der Elemente <solution> und <how> Informationen, die bei PLML 1.1 dem Element <solution> zuzuordnen sind. Darüber hinaus liefert <how> gelegentlich Abbildungen oder Skizzen, die in das PLML 1.1-Element <diagram> eingegeben werden können. Informationen hinsichtlich <related-patterns> bei PLML 1.1 können aus <use-when>, <how> und <why> entnommen werden. Die Elemente <patternID> und <implementation> sind nicht berücksichtigt.

Die Muster in der Sammlung *Design of Sites* von van Duyne verfügen über einen, dem PLML 1.1-Element <patternID> entsprechenden, eindeutigen Bezeichner. Er besteht aus einem Großbuchstaben, der die Kategorie des Patterns kennzeichnet und einer fortlaufenden Nummer. Die Inhalte des Elements <context> bei PLML 1.1 können aus Informationen aus <background> und <problem> zusammengesetzt werden. Das Element <problem> bei van Welie umfasst eine Vielzahl von Informationen, die gleich mehreren PLML 1.1-Elementen zugeordnet werden können: <problem>, <context>, <evidence>, <implementation> und <related-patterns>. Es sind keine Informationen im Zusammenhang mit dem PLML 1.1-Element <literature> vorhanden.

Bei der *Quince Pattern Library* lassen sich die dort vorhandenen Beschreibungselemente fast vollständig eindeutig PLML 1.1-Elementen zuordnen. Einzig erfolgt die Beschreibung der Beziehungen zu anderen Mustern nicht explizit über Bezeichner oder Namen der betreffenden Patterns mittels <pattern-link> oder <related-patterns>, sondern implizit mithilfe von im Element <tags> gespeicherten Kennzeichen. Nicht adressiert werden die PLML 1.1-Elemente <patternID>, <alias>, <illustration>, <forces>, <synopsis>, <confidence>, <pattern-link> und <management>.

Bei der *Yahoo Design Pattern Library* enthält das Beschreibungselement <illustration> gelegentlich auch Abbildungen, die dem PLML 1.1-Element <diagram> zuzuordnen sind. Informationen zur Implementierung sind gleichermaßen in den Beschreibungselementen <accessibility>, <special-cases> und <code-examples> sowie gelegentlich auch in <solution> enthalten. Beispiele für die Anwendung der Muster sind in den Elementen <examples>, <as-used-on-Yahoo>, <other-examples> und <pattern-gallery> untergebracht. Beziehungen zwischen den Patterns werden mittels <related-patterns>, <similar-patterns-in-other-libraries> und <disambiguation> dokumentiert. Neben den bereits weiter oben genannten PLML 1.1-Elementen ohne Entsprechungen sind zudem keine Informationen hinsichtlich <patternID> und <literature> verfügbar.

Der vollständige Vergleich der Beschreibungselemente der betrachteten Pattern-Sammlungen und deren Zuordnung zu PLML 1.1 befindet sich in Anhang C.6.

Abschließend lässt sich feststellen, dass wegen der Existenz von mehrdeutigen Zuordnungsbeziehungen eine vollständig automatisierte Transformation der in den untersuchten Muster-Sammlungen enthaltenen Pattern-Spezifikationen in das PLML 1.1-Format nicht durchgeführt werden kann.

3.4. Pattern-Werkzeuge

Im Rahmen der Analysen zur vorliegenden Dissertation wurden in einem Literatur-Review neun Pattern-Werkzeuge mittels Literatur-Review und zusätzlich fünf Webseiten zu Pattern-Sammlungen insbesondere auf die jeweils angebotenen Funktionalitäten untersucht. Im folgenden Unterkapitel 3.4.1 werden einige grundsätzliche Begriffe definiert. In Unterkapitel 3.4.2 werden dann folgende Tools zusammenfassend beschrieben: *Usability Pattern-Assisted Design Environment* (UPADE), *Damask*, *Montreal Online Usability Patterns Digital Library* (MOUDIL), *Management of User Interface Patterns* (MUIP), *HCI Pattern Language Management Tool* (HCI PLMT), *Patterns in Modelling* (PIM) *Tool*, *Task Pattern Wizard* und *PatternWiki*. Aufgrund ihrer sehr ähnlichen Funktionalitäten erfolgt die Beschreibung der

Website-Tools *Designing Interfaces*, *Patterns in Interaction Design*, *Design of Sites*, *Quince UX Patterns Explorer* und *Yahoo Design Pattern Library* gemeinsam in Unterkapitel 3.4.2.10.

Abbildung 7 gibt eine Übersicht über die zeitliche Einordnung der Pattern-Werkzeuge. Da für die Website-Tools die chronologische Entwicklung nicht oder nicht gesichert ermittelbar war, sind sie in der Abbildung nicht dargestellt. Hinsichtlich des Werkzeugs *Damast* lagen zwei zeitlich länger auseinanderliegende Veröffentlichungen aus den Jahren 2002 und 2008 vor. Dieser Sachverhalt wird durch die gestrichelte Linie dargestellt.

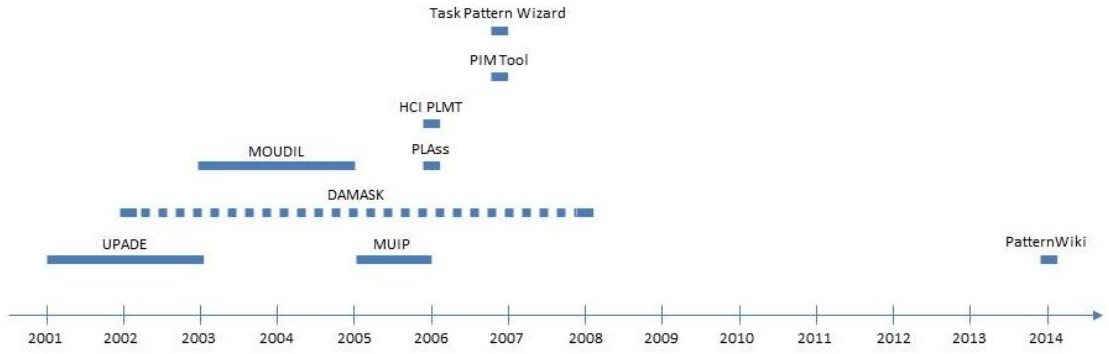


Abbildung 12 ZEITLICHE EINORDNUNG DER UNTERSUCHTEN PATTERN-WERKZEUGE

Anschließend erfolgen in Unterkapitel 3.3.3 der Vergleich und die Bewertung dieser Werkzeuge anhand eines zuvor festgelegten Schemas. Insbesondere wurden die formalen Pattern-Beschreibungen und die jeweils vorhandene Unterstützung der Benutzer hinsichtlich des Pattern-Managements, der kollektiven Zusammenarbeit und der Anwendung von Mustern untersucht.

3.4.1. Begriffsdefinitionen

3.4.1.1. Pattern-Katalog-Werkzeug

Ein Pattern-Katalog-Werkzeug (engl. Pattern Catalogue Tool) präsentiert seinen Benutzern gespeicherte Muster, die in Kategorien organisiert sind. Das Hauptanliegen dieser Art von Werkzeugen ist eine hochgradige Verbreitung von Design-Know-how. Der Zugriff erfolgt deshalb auch häufig über das Internet. Mitunter unterstützen Katalog-Tools auch die Anlage neuer Patterns bzw. Einreichung von Vorschlägen hierfür [187].

3.4.1.2. Pattern-Management-Werkzeug

Ein Pattern-Management-Werkzeug (engl. Pattern Management Tool) konzentriert sich auf das Anlegen und die Organisation von Mustern in Pattern-Sammlungen. Die Erstellung neuer Patterns erfolgt auf Basis von Templates, um ein einheitliches Beschreibungsformat zu erreichen. Es sind Such und Browse-Funktionalitäten vorhanden und es besteht die Möglichkeit, sich die Patterns aus verschiedenen Blickwinkeln anzeigen zu lassen. Zudem können bereits vorhandene Muster modifiziert und Beziehungen zwischen diesen definiert werden [187].

3.4.1.3. Pattern-basiertes Entwurfs-Werkzeug

Ein Pattern-basiertes Entwurfs-Werkzeug (engl. Pattern-based Design Tool) unterstützt seine Benutzer bei der Nutzung und Anwendung von Mustern im Rahmen von, teilweise auch automatisierten, Entwurfsaktivitäten. Gelegentlich sind auch rudimentäre Funktionalitäten hinsichtlich der Neuanlage und der Änderung von Mustern vorhanden [187].

3.4.2. Untersuchte Pattern-Werkzeuge

3.4.2.1. UPADE

Das *Usability Pattern-Assisted Design Environment* (UPADE) wurde an der *Concordia University* in Montreal (Kanada) entwickelt. Die erste Veröffentlichung stammt aus dem Jahr 2001 [188]. Neben Möglichkeiten zum Management von Patterns bietet UPADE auch Unterstützung hinsichtlich der Anwendung von Patterns beim Entwurf von Benutzeroberflächen und der automatisierten Generierung von Quell-Code an [188] [189]. Hierzu wird bei UPADE grundsätzlich zwischen den drei Abstraktionsebenen der Pattern-, Design- und Code-Ebene unterschieden [189].

Die Beschreibung der Patterns erfolgt auf Basis von PCML (siehe Kapitel 3.2.1.5). Den Benutzern von UPADE stehen Funktionen zur Erfassung, Änderung, Recherche (Browsing) und Suche von Patterns sowie zur Definition von Beziehungen zwischen den Patterns zur Verfügung [189]. Durch die Verwendung von *Cascading Stlye Sheets*[166] (CSS) und *Extensible Stylesheet Language*[167] (XSL) können diese Funktionalitäten durch Transformationen in *Extensible Hyper Text Markup Language*[168] (XHTML), *Scalable Vector Graphics*[169] (SVG), *Wireless Markup Language*[170] (WML) oder *Voice Extensible Markup Language*[171] (VoiceXML) auf verschiedenen Endgeräten zur Verfügung gestellt werden. Darüber hinaus umfasst UPADE einen graphischen Editor für den Entwurf von Benutzeroberflächen durch Komposition der verfügbaren Patterns [188] [189]. Beispiele für Patterns und deren Anwendung in Pattern-Kompositionen hinsichtlich der Erstellung von Web-Seiten sind in [188] beschrieben. In einer prototypischen Implementierung wird versucht, Klassen-Definitionen und Quell-Code-Fragmente aus den Inhalten der Pattern-Beschreibungen zu generieren [189].

3.4.2.2. Damask

Der erste Entwurf des Pattern-Werkzeugs *Damask*[172] wurde an der *University of California* in Berkeley (USA) im Jahr 2002 erstellt [190]. Die Implementierung erfolgte offensichtlich erst

---

[166] Siehe http://www.w3.org/Style/CSS/
[167] Siehe http://www.w3.org/Style/XSL/
[168] Siehe http://www.w3schools.com/html/html_xhtml.asp
[169] Siehe http://www.w3.org/Graphics/SVG/
[170] Siehe http://www.wirelessdevnet.com/channels/wap/training/wml.html
[171] Siehe http://www.w3.org/TR/voicexml30/
[172] Engl.: Damast; ein dicker, häufig auch glänzender Stoff mit eingewebten Mustern (siehe http://www.merriam-webster.com/dictionary/damask)

deutlich später im Jahr 2008 als gemeinschaftliche Arbeit des *IBM Almaden Research Center* in San José (USA) und der *University of Washington* in Seattle (USA) [191]. Ziel von Damask ist die zeitgleiche Erstellung Sketch-basierter User Interface-Entwürfe für unterschiedliche Arten von Endgeräten. So wird der Entwurf von Benutzeroberflächen mit Interaktionsmöglichkeiten im Web-Stil für Desktop-PC und Mobiltelefone sowie sprachbasierter User Interfaces für Telefone oder PC im sogenannten Prompt-and-Response[173]-Verfahren unterstützt [190] [191].

UI-Designer skizzieren Benutzeroberflächen mittels digitalen Stifts. Hierzu stehen Ihnen grafische Elemente wie Seiten (Pages), Kontroll-Elemente (Controls) und Pfeile (Arrows) zur Verfügung. Seiten repräsentieren hierbei Web-Seiten oder Bildschirme, Controls die Inhalte der Seiten in Form von verschiedenen UI-Elementen und Pfeile die Beziehungen zwischen den Seiten. Die Sprach-Schnittstellen bestehen aus Aufforderungen (Prompts), Antworten (Responses) und Formularen (Forms). Die Formulare dienen dazu, zusammengehörende Prompts und Responses zu gruppieren. Der UI-Entwurf erfolgt in verschiedenen Schichten (Layers). In einer allgemeinen Schicht werden alle UI-Elemente platziert, die auf allen Endgeräten gleichermaßen erscheinen sollen. Zudem gibt es für jedes unterstützte Device jeweils eine geräteabhängige Schicht, die alle UI-Elemente enthält, die nur auf dem entsprechenden System zur Verfügung stehen [191].

Um die Erstellung der UI-Entwürfe zu vereinfachen, zu beschleunigen und die Konsistenz der User Interfaces für die verschiedenen Typen von Endgeräten zu erhöhen, kommen Patterns zum Einsatz [190] [191]. Damask enthält 90 Muster aus der Pattern-Sammlung *Design of Sites* (siehe Kapitel 3.3.2.3), die gemäß *PLML Version 1.1* (siehe Kapitel 3.2.1.1) strukturiert und gespeichert sind. Hiervon wurden elf Muster um vordefinierte UI-Design-Fragmente, die ebenso aus den vorher genannten grafischen Elementen bestehen, ergänzt [191]. Diese Fragmente, von denen für jedes unterstützte Endgerät jeweils eines existiert, sind im Beschreibungselement <solution> untergebracht. Wird ein Pattern ausgewählt und angewendet, so werden die Fragmente simultan in die entsprechenden geräteabhängigen UI-Designs eingefügt und können aber anschließend noch manuell verändert werden. Die letztlich im UI-Design verwendeten Varianten der Fragmente werden in das Beschreibungselement <example> des betreffenden Patterns zurückgeschrieben [190].

3.4.2.3. MOUDIL

Auch die *Montreal Online Usability Patterns Digital Library* (MOUDIL) wurde an der *Concordia University* in Montreal (Kanada) entwickelt. Die erste Veröffentlichung erfolgte im Jahr 2003 [192]. Erklärtes Ziel von MOUDIL ist die Bereitstellung eines umfassenden Frameworks zum Austausch und zur Verbreitung von Design-Know-how in Form von HCI Patterns [192] [193], sowie zu deren automatisierten Anwendung [193].

Primär wird das Ziel verfolgt, Patterns einheitlich zu beschreiben, in einem zentralen Repository zu verwalten und für alle Interessierten zugänglich zu machen. Hierfür wurde die soge-

[173] Engl.: Aufforderung und Antwort. Bei *Prompt-and-Response*-Schnittstellen erfolgt die Kommunikation zwischen Mensch und Maschine durch gesprochene Sprache. Durch einen durch das System gesprochenen Prompt wird der Benutzer zu einer Antwort (Response) aufgefordert.

nannte *Seven C's Methodology* definiert, deren einzelne Schritte durch das Werkzeug unterstützt werden sollen [192] [193] [194]:

1. Collect (Sammeln)
   Konzentration der verstreut vorhandenen Informationen über Patterns an einem zentralen Ort.
2. Cleanup (Säubern)
   Vereinheitlichung der Repräsentation der Patterns.
3. Certify (Bescheinigen)
   Definition von Domäne und Terminologie für die Pattern-Sammlung.
4. Contribute (Beitragen)
   Gemeinschaftliche, ständige Erweiterung und Verbesserung der Pattern-Sammlung.
5. Categorize (Kategorisieren)
   Definition von Kategorien bzw. einer Hierarchie von Kategorien, um die Verwaltung und Verwendung der Pattern-Sammlung zu vereinfachen.
6. Connect (Verbinden)
   Erkennung und Dokumentation der zwischen den Patterns bestehenden Beziehungen.
7. Control (Kontrolle)
   Definition eines maschinenlesbaren Formats, das die automatisierte Verarbeitung der Patterns durch Software-Werkzeuge ermöglicht.

Die Beschreibung der Patterns erfolgt mittels XML. Das zugehörige Schema sieht eine Gruppierung von Informationen anhand der Beschreibungselemente Head, Body, Relations und Assimilation vor. Hierbei vereint [193]:

- Head (Kopf)
  die Meta-Daten des Patterns, z. B. Name, eindeutiger Bezeichner und Autor.
- Body (Körper)
  intrinsische Informationen, z. B. Problem-, Kontext- und Lösungsbeschreibungen sowie Anwendungsbeispiele.
- Relations (Beziehungen)
  extrinsische Informationen, also die Beziehungen mit anderen Patterns.
- Assimilation (im Sinne von Nutzung)
  Angaben darüber, wann, wo und wie das betreffende Pattern verwendbar ist sowie die für die automatische Anwendung und/oder Code-Generierung benötigten Informationen.

Generell werden die Rollen des Autors und des Benutzers von Patterns unterschieden, bei denen potenzielle Unterschiede hinsichtlich Vorwissen und Fähigkeiten unterstellt werden können. Daraus entstehende Verständigungsprobleme sollen durch die oben beschriebene Methodik und damit durch MOUDIL vermieden bzw. ausgeglichen werden [193] [194]. Zudem wird die Idee eines offenen, internationalen Redaktionsgremiums, das vorgeschlagene Patterns vor deren Veröffentlichung prüft und anerkennt, verfolgt [192] [193].

3.4.2.4. MUIP

Das Pattern-Tool *Management of User Interface Patterns* (MUIP) wurde an der *Massey University* in Palmerston North (Neuseeland) entwickelt. Die erste Veröffentlichung erfolgte im Jahr 2005 [187].Wie der Name des Werkzeugs bereits nahelegt, liegt der Fokus bei MUIP

auf den Aspekten des Pattern-Managements. Insbesondere sollen die MUIP-Benutzer bei der Erstellung eigener Pattern-Sammlungen unterstützt werden und dabei bereits veröffentlichtes Wissen auf einfache Art und Weise integrieren können [174].

Hierfür wurden folgende Funktionalitäten definiert und implementiert [174] [187]:

1. Erstellung von Patterns (Pattern Authoring)
   Alle Aktivitäten, die die Erstellung von Pattern-Beschreibungen betreffen, z. B. das Befüllen der Beschreibungselemente mit Inhalten, das Einfügen von Multimedia-Daten oder Code-Fragmenten in die Pattern-Beschreibungen und die Erstellung von Vorlagen.
2. Ändern der Einflüsse (Manipulating Forces)
   Ändern oder Erweitern der Einflüsse, die in der Liste im Beschreibungsfeld <forces> der jeweiligen Patterns verwaltet werden.
3. Recherche (Browsing)
   Anzeigen verschiedener Pattern-Listen und -Ansichten (Übersichten und Details).
4. Suche (Searching)
   Suche von Patterns mittels Schlüsselworten, Angabe von Beschreibungsfeld-Inhalten oder Volltext-Suche.
5. Änderung (Modification)
   Veränderung und Anpassung von Pattern-Beschreibungen mithilfe einer Versionskontrolle, Erstellung von Anmerkungen, Bewertung von Patterns.
6. Definieren von Beziehungen (Relating Patterns)
   Erstellen von Beziehungen zwischen Patterns, Definition neuer Beziehungs-Typen, Visualisierung von Beziehungen.
7. Beeinflussen von Pattern-Sammlungen (Manipulating Compilations)
   Erstellen neuer Pattern-Sammlungen für den persönlichen Gebrauch, Erstellen von Pattern-Kategorien, Aufnehmen von Patterns aus anderen Sammlungen in die eigene Sammlung.
8. Ein- und Ausgabe (Input and Output)
   Import- und Export einzelner Patterns, ganzer Pattern-Sammlungen oder Pattern-Übersichten, Transformation in verschiedene Beschreibungsformate.

Die Beschreibung der Patterns erfolgt auf Basis der Beschreibungssprache PLML 1.2 (siehe Kapitel 3.2.1.2), die eigens für die Implementierung von MUIP entwickelt wurde. Unter anderem liegt besonderer Augenmerk auf dem Beschreibungselement <forces>, das dazu verwendet werden kann, die Beziehungen zwischen Patterns zu untersuchen und passende Patterns im Rahmen des Entwurfs von Benutzerschnittstellen auszuwählen.

3.4.2.5. PLAss

Der *Pattern Language Assistant* (PLAss) wurde im Rahmen der Entwicklung einer Pattern-Sprache für die Gestaltung von Websites an der *Hochschule Augsburg* [174] (Deutschland) entwickelt. Die Veröffentlichung erfolgte im Jahr 2006 [195].

[174] Damaliger Name *Fachhochschule Augsburg*

Bei PLAss wird zwischen zwei verschiedenen Betriebs-Modi unterschieden. Während ein als „offen“ bezeichneter Modus die Änderung und Anpassung jeglicher Inhalte erlaubt, können bei der schreibgeschützten Betriebsart die Inhalte ausschließlich angezeigt werden. Auf diese Weise werden praktisch die zwei Rollen des Autors und des Nutzers von Patterns bzw. der betreffenden Pattern-Sprache realisiert [195].

Die Spezifikation der Muster erfolgt nicht auf Basis einer fest vorgegebenen Beschreibungssprache, sondern es besteht die grundsätzliche Möglichkeit, die zum Einsatz kommenden Beschreibungselemente für jede zu definierende Pattern-Language individuell festzulegen. Darüber hinaus verfügt das Werkzeug über Such- und Browsing-Funktionen. Im offenen Modus können zudem neue Patterns angelegt und die Beschreibungen von bestehenden Mustern jederzeit modifiziert werden. Bei der Spezifikation von Beziehungen zwischen den Patterns erfolgt eine automatisierte Unterstützung des Benutzers durch eine Kontrollinstanz, die auf einem Regelwerk zur Erstellung von Pattern-Hierarchien basiert. Die Speicherung sowohl der Struktur als auch der Inhalte der Muster erfolgt XML-konformen Dateien. Aus diesem Grund kann ein Export in anderen Formaten mittels XSLT[175] realisiert werden. PLAss unterstützt hierbei standardmäßig bereits HTML, PDF und das DOC-Format von Microsoft Office. Ein Import kann außerhalb des Tools durch Manipulation der XML-Dateien bewerkstelligt werden [195].

3.4.2.6. HCI PLMT

Das *HCI Pattern Language Management Tool* (HCI PLMT) wurde im Jahr 2006 an der *American University in Cairo* (Ägypten) entwickelt. Es handelt sich um eine Web-Anwendung, die ihre Benutzer bei der Recherche und Suche von Patterns unterstützt und die Einbindung bereits existierender Patterns sowie die Definition von Beziehungen zwischen den Patterns erlaubt. Auf eine Lösch-Funktion für Patterns wurde verzichtet; ebenso ist das Anlegen neuer Patterns nicht möglich [196].

Die Beschreibung der Patterns erfolgt mittels obligatorischer Felder für Pattern-Name, Problem, Lösung, Web-Site-Link, Pattern-Kategorie, ein optionales Feld für den Kontext und weitere nicht-verpflichtende Felder, zu denen keine weitere Detailinformationen vorliegen. Im Feld Web-Site-Link befindet sich der *Uniform Resource Locator* (URL) der tatsächlichen Pattern-Definition. Für die Spezifikation von Beziehungen zwischen Patterns werden die Inhalte der Felder für Pattern-Namen, Problem, Kontext und Lösung automatisch ausgewertet und darin durch String-Vergleiche nach den Namen anderer Patterns gesucht. Ergeben sich Treffer bei den Vergleichs-Operationen, so werden diese als potenzielle Beziehung zwischen den betreffenden Patterns interpretiert. Potenzielle Alternativen zu einem Pattern werden daran erkannt, dass sie über den jeweils gleichen Pattern-Namen verfügen, aber verschiedene Web-Site-Links besitzen. Für jede ausgewählte Beziehung wird der jeweilige Link-Typ festgelegt. Mögliche Typen sind hierbei Assoziation, Aggregation, Spezialisierung und Alternative. Es besteht die Möglichkeit, den Typ einer Beziehung im Nachhinein zu verändern [196].

[175] Siehe http://www.w3.org/TR/xslt

3.4.2.7. PIM Tool

Das *Patterns in Modelling* (PIM) *Tool* ist eine gemeinschaftliche Entwicklung der *Universität Rostock* (Deutschland) und der *Concordia University* in Montreal (Kanada) aus dem Jahr 2007 [20]. Ziel dieses Werkzeugs ist es, modellgetriebene und musterbasierte Entwicklungsansätze zu kombinieren. Modellseitig wird die Erstellung eines sogenannten *UI Multi Model* angestrebt, das intern aus je einem Aufgaben-, Dialog-, Präsentations- und Layout-Modell besteht. Hierbei dienen die Task-, Dialog- und Präsentations-Modelle den in Kapitel 2.2 beschriebenen Zwecken. Das Layout-Modell ermöglicht die Zuweisung von Stil-Attributen, wie beispielsweise Größe, Schriftart oder Farbgebung, zu den abstrakten UI-Elementen des Präsentationsmodells. Es stehen geeignete Patterns zur Verfügung, aus denen die genannten vier Teilmodelle bausteinartig zusammengesetzt und erweitert werden können. In einer ersten Ausbaustufe des Werkzeugs kann die Erstellung von Aufgaben-Modellen mithilfe von Patterns, die um Task-Modell-Informationen angereichert wurden, vereinfacht und beschleunigt werden [20].

Die Beschreibung der Patterns erfolgt auf Basis der Beschreibungssprache TPML (siehe Kapitel 3.2.1.6). Das PIM Tool unterstützt die Anwendung der Muster, die in einem Prozess mit folgenden vier Schritten erfolgt [20]:

1. Identifikation (Identification)
   Es wird ein Teilmodell des zu erstellenden Modells identifiziert, das mithilfe eines Patterns ersetzt oder erweitert werden soll.
2. Auswahl (Selection)
   Es wird ein geeignetes Pattern ausgewählt, das der gewünschten Ersetzung oder Erweiterung entspricht.
3. Instanziierung (Instantiation)
   Es wird die dem Anwendungskontext entsprechende Struktur des Patterns hergestellt, indem die variablen Anteile des Patterns (siehe Kapitel 3.2.1.6) mit konkreten Werten versehen werden.
4. Integration
   Die im vorhergehenden Schritt spezifizierte Pattern-Struktur wird in das Zielmodell integriert und benötigte Verbindungen zu den anderen Teilmodellen hergestellt.

Das PIM Tool unterstützt ausschließlich die Erstellung des *UI Multi Model* durch die Verwendung von Patterns, die praktisch als Modell-Bausteine dienen. Die schrittweise Generierung einer ablauffähigen Benutzeroberfläche aus diesem Modell erfolgt außerhalb der Funktionalität des Werkzeugs. Hierfür wird beispielsweise eine geeignete modellbasierte UI-Entwicklungsumgebung benötigt [20].

3.4.2.8. Task Pattern Wizard

Der *Task Pattern Wizard* wurde an der *Concordia University* in Montreal (Kanada) entwickelt. Die Veröffentlichung erfolgte im Jahr 2007 [197]. Das Werkzeug basiert auf denselben Ansätzen wie das zuvor beschriebene *PIM Tool* (siehe Kapitel 3.4.2.7), an dessen Entwurf die *Concordia University* beteiligt war. Die beiden Werkzeuge verbinden modellgetriebene und musterbasierte UI-Entwicklungsmethoden und unterstützen mit Aufgaben-, Dialog-, Präsentations- und Layout-Modellen dieselben Modelltypen. Insbesondere imple-

mentieren sie die gleichen Mechanismen hinsichtlich der Anwendung von Patterns in den vier Schritten Identifikation, Auswahl, Instanziierung und Integration [197].

Die Erstellung der Modelle erfolgt in dem durch den *Task Pattern Wizard* unterstützten modellgetriebenen Prozess in drei Phasen. In einem ersten Schritt werden jeweils ein Domänen-, Benutzer- und Aufgaben-Modell definiert. In der folgenden Phase erfolgt die Erstellung von Umgebungs- (Environment), Dialog- und Präsentations-Modellen. Hierbei umfasst das Environment-Modell neben den physischen Gegebenheiten des Aufenthaltsortes des Benutzers auch die Spezifikation der zum Einsatz kommenden Endgeräte-Plattform. In der abschließenden dritten Phase wird das Layout-Modell festgelegt. Bei diesem Vorgehen erfolgt eine schrittweise Konkretisierung der Benutzeroberfläche, bis schließlich aus dem Layout-Modell das eigentliche User-Interface generiert werden kann [197].

Gemäß dieses Prozesses werden Patterns beim *Task Pattern Wizard* für die Erfüllung von zwei unterschiedlichen Aufgaben verwendet. Einerseits dienen sie als Bausteine für die Konstruktion der eingangs erwähnten vier Teilmodelle, andererseits unterstützen Patterns auch die erforderlichen Modell-Transformationen [197].

Die Beschreibungen der Patterns enthalten zumindest die Informationen hinsichtlich der jeweiligen Problemstellung, des Kontextes und der vorgeschlagenen Lösung des Problems. Zudem wird die Eignung der Lösung begründet. Die in den Patterns enthaltenen Modell-Fragmente werden mithilfe der *XML User Interface Language* (XUL, siehe Kapitel 2.4.1.2) spezifiziert [197].

3.4.2.9. PatternWiki

PatternWiki ist eine gemeinschaftliche Entwicklung der *Rheinisch-Westfälischen Technischen Hochschule (RWTH) Aachen* und der *Universität Duisburg-Essen*. Die erste Veröffentlichung fand im Jahr 2014 statt. Bei der Entwicklung des Werkzeugs standen insbesondere Aspekte der Kollaboration hinsichtlich der Erstellung und Pflege von Patterns im Vordergrund [198].

Basis von PatternWiki ist das frei verfügbare Softwarepaket *MediaWiki*[176], das um zwei Erweiterungen ergänzt wurde. Einerseits handelt es sich dabei um eine Erweiterung zur Verarbeitung von XML-konformen Datenquellen und andererseits um eine grafische Erweiterung. Die XML-Extension erlaubt die Transformation von XML-basierten Pattern-Beschreibungen in das von der MediaWiki-Engine darstellbare Datenformat namens *WikiText*[177] . Zudem können damit HTML-Formulare aus XML-Quellen generiert werden, um den Benutzern von PatternWiki die Eingabe und Änderung von Pattern-Beschreibungen zu erleichtern. Die grafische Erweiterung erlaubt es, die definierten Beziehungen zwischen den Patterns anhand von Graphen zu visualisieren [198].

Die Beschreibung der Patterns erfolgt gemäß der Beschreibungssprache PLML (siehe Kapitel 3.2.1.1) und kann durch die Verwendung gebräuchlicher XML-Techniken wie *Extensible Stylesheet Language Transformations*[178] (XSLT) und *XML Schema Definition*[179]

[176] Siehe http://www.mediawiki.org/wiki/MediaWiki
[177] Siehe http://www.mediawiki.org/wiki/Wikitext
[178] Siehe http://www.w3.org/TR/xslt
[179] Siehe http://www.w3.org/XML/Schema

(XSD) auch in andere Formate umgewandelt werden. PatternWiki unterstützt die Erfassung und Änderung sowie die Recherche (Browsing) und gezielte Suche von Patterns. Ferner können Beziehungen zwischen den Patterns definiert werden. Dank der Verwendung von MediaWiki stehen Kollaborations-Funktionalitäten wie Diskussion, Kommentierung und Revision für die Benutzer zur Verfügung. Auf Basis von *XSLT Style Sheets* kann eine automatisierte Generierung von Quell-Code oder von anderen Datenformaten, die beispielsweise durch MB-UIDEs (siehe Kapitel 2.5) weiter verarbeitet werden können, aus den Inhalten der Pattern-Beschreibungen erfolgen [198].

3.4.2.10. Web-Seiten zu Pattern-Sammlungen

Die Online-Versionen der in den Kapiteln 3.3.2.1 bis 3.3.2.5 beschriebenen Pattern-Sammlungen verfügen über Browser-basierte User-Front-Ends, die durchaus als, wenngleich auch meist verhältnismäßig einfache, Pattern-Werkzeuge angesehen werden können.

Das Online-Werkzeug *Designing Interfaces* zur Pattern-Sammlung von Jenifer Tidwell steht in zwei Varianten zur Verfügung. Die ältere Version [199] umfasst weniger Patterns als die neue, stellt aber nur einfache Browsing- und keinerlei Such-Möglichkeiten zur Verfügung. Mit der neuen Version [182] können die Benutzer hingegen sowohl Browsen als auch Suchen. Zudem sind in einer statischen Überblicksansicht zusätzliche Patterns enthalten, jedoch werden nur zu 18 dieser Muster die Detailinformationen tatsächlich angeboten. In beiden Varianten steht ein Navigation-Panel zur Verfügung, in dem die verfügbaren Muster pro Kategorie angezeigt werden. Details zur Organisation der zugehörigen Pattern-Sammlung sind in Kapitel 3.3.2.1 zu finden. Das letzte Update der Webseite erfolgte im Jahr 2011 [182].

Durch das Front-End zur Pattern-Sammlung *Patterns in Interaction Design* von Martijn van Welie [183] werden eine Ansicht mit einem Überblick über die vorhandenen Patterns pro Kategorie sowie Funktionalitäten zum Browsen und Suchen angeboten. Zudem können die Benutzer Kommentare zu den Mustern hinterlassen und die Bemerkungen anderer User lesen. Darüber hinaus beinhaltet die Webseite Verweise auf andere Pattern-Sammlungen in Form von Links und es können sowohl Nachrichten an den Pattern-Autor gesendet als auch neue Muster unter der Angabe von dessen Namen, einer Beschreibung und Anwendungsbeispielen vorgeschlagen werden. Details zur Organisation der Muster sind in Kapitel 3.3.2.2 enthalten. Die letzte Aktualisierung der Webseite erfolgte im Jahr 2008 [183].

Einen sehr ähnlichen Funktionsumfang weist das User-Interface von *Design of Sites* von Douglas van Duyne auf. Es existieren hier fast identische Möglichkeiten zum Anzeigen der vorhandenen Patterns pro Kategorie und zum Browsen und Suchen. Die Seite hält aber zusätzlich generelle Hintergrundinformationen zu Mustern bereit und die Benutzer können den Pattern-Beschreibungen eigene Anwendungsbeispiele hinzufügen. Die Organisation der zugehörigen Pattern-Sammlung ist in Kapitel 3.3.2.3 beschrieben. Das letzte Update der Web-Präsenz fand im Jahr 2009 statt [184].

Ebenfalls ähnlich bezüglich der Funktionen zum Browsen und Suchen von Mustern ist das Web-Front-End der *Yahoo Design Pattern Library* der Firma *Yahoo! Inc.* [185]. Ein

Navigation-Panel zeigt alle verfügbaren Muster pro Kategorie. Mittels Verlinkungen zu den Blogs und Foren des Yahoo Developer Network[180] (YDN) haben die Benutzer die Möglichkeit, an Diskussionen über die Patterns teilzunehmen. Darüber hinaus umfasst die Webseite sehr ausführliche Hintergrundinformationen zu Mustern, Links auf andere Pattern-Sammlungen sowie eine Auflistung der unlängst aktualisierten Pattern-Beschreibungen. Außerdem können Drahtmodell-Schablonen (engl. Wire Frame Stencils) zur Verwendung im Design-Prozess, z. B. bei der Erstellung von Skizzen, heruntergeladen werden. Die Organisation der Muster befindet sich in Kapitel 3.3.2.4. Hinsichtlich der Aktualität des Online-Werkzeugs konnten keine Informationen aufgefunden werden.

Den *Quince UX Patterns Explorer* der Firma *Infragistics Inc.* [186] kann in mehreren Varianten bzw. Konfigurationen verwendet werden. Die einfachste ist hierbei eine öffentliche Version, bei der alle vorhandenen Patterns in einem alphabetisch nach Namen sortierten Verzeichnis aufgelistet sind und durch Anklicken jeweils die Details zu den jeweiligen Mustern angezeigt werden. Mehr Funktionalitäten werden angeboten, wenn auf dem verwendeten Endgerät das Produkt *Silverlight* [181] von der Firma *Microsoft Corp.* installiert und aktiviert ist. Hierbei stehen dann sowohl Such- als auch verschiedene Browsing-Möglichkeiten zur Verfügung. Ist man zusätzlich als registrierter Benutzer bei der Anwendung angemeldet, erweitert sich das Funktionsspektrum noch einmal und man bekommt die jüngsten Neuerungen angezeigt, kann Kommentare zu Patterns hinterlassen, Anwendungsbeispiele diskutieren (im Tool als *Corkboard*-Funktion bezeichnet), Nachrichten an andere registrierte User versenden und neue Patterns vorschlagen. Die kostenpflichtige Version *Quince Pro* beinhaltet schließlich ergänzend private Speicherbereiche (sogenannte *Design Libraries*) für eigene Anpassungen und eine *Design Boards* genannte Funktion, mit der man Patterns automatisiert bei der Modellierung von Aktionsabfolgen für eine Benutzeroberfläche anwenden kann. Informationen zur Organisation der Pattern-Sammlung befinden sich in Kapitel 3.3.2.5. Auch für den *Quince UX Patterns Explorer* konnten keine Angaben zur Aktualität der Anwendung ermittelt werden.

3.4.3. Vergleich und Bewertung

Es wurden bereits Studien zu Pattern-Werkzeugen durchgeführt und veröffentlicht, beispielswiese [187] und [145]. Diese stammen jedoch aus den Jahren 2005 bzw. 2009 und deshalb bleiben einige sehr interessante, neuere Ansätze unberücksichtigt. Nichtsdestotrotz lieferten diese Dokumente wertvolle Informationen, insbesondere hinsichtlich der zu untersuchenden Eigenschaften der Tools.

Alle zuvor beschriebenen Pattern-Werkzeuge wurden auf folgende Merkmale untersucht:

1. Name des Werkzeugs
   a. Kurzbezeichnung
   b. Vollständiger Name (falls zutreffend)
2. Urheber
3. Datum des Entwicklungsbeginns bzw. der Erstveröffentlichung

[180] Siehe https://developer.yahoo.com/
[181] Siehe http://www.microsoft.com/silverlight/

4. Datum der Freigabe der aktuellsten Version bzw. der aktuellsten Publikation
5. Zugehörigkeit zu einer Werkzeug-Kategorie (siehe Definitionen in Kapitel 3.4.1)
   a. Pattern-Katalog-Werkzeug
   b. Pattern-Management-Werkzeug
   c. Pattern-basiertes Entwurfs-Werkzeug
6. Unterstützte Pattern-Beschreibungssprache bzw. -Beschreibungselemente
7. Unterstützte Pattern-Management-Funktionen (Details siehe [187])
   a. Authoring
   b. Browsing
   c. Suche
   d. Modifikation
   e. Definition von Beziehungen zwischen Patterns
   f. Manipulation von Pattern-Sammlungen bzw. privater Speicherbereich
   h. Im- und Export-Möglichkeiten
8. Unterstützte Kollaborations-Funktionen
   a. Diskussion bzw. Meinungsaustausch
   b. Review und Kommentierung von Pattern-Beschreibungen
   c. Überarbeitung von Pattern-Beschreibungen inkl. Change-Log
9. Unterstützte Funktionalitäten zur (automatisierten) Anwendung von Patterns
10. Grundlegende Konzepte, auf denen das Werkzeug basiert
11. Verfügbarkeit des Werkzeugs
    a. Im Sinne der grundsätzlichen Existenz einer ablauffähigen Anwendung
    b. Im Sinne eines tatsächlich möglichen Zugriffs auf das Werkzeug
12. Art und Anzahl der vorhandenen Quellen und Dokumentationen

Ein Überblick über die im Literaturreview berücksichtigten und ausgewerteten Quellen wird in Tabelle 15 gegeben.

Tabelle 15 IM LITERATURREVIEW BERÜCKSICHTIGTE QUELLEN ZU DEN PATTERN-WERKZEUGEN

Pattern-Werkzeug	Quellen
UPADE	[188] [189]
Damask	[190] [191]
MOUDIL	[192] [193] [194]
MUIP	[174] [187]
PLAss	[195]
HCI PLMT	[196]
PIM Tool	[20]
Task Pattern Wizard	[197]
PatternWiki	[198]
Designing Interfaces	[182] [199]
Patterns in Interaction Design	[183]
Design of Sites	[184]
Yahoo Design Pattern Library	[185]
Quince UX Patterns Explorer	[186]

Das Usability Pattern-Assisted Design Environment (UPADE) bietet bezüglich des Managements von Patterns vielfältige Funktionalitäten. Ein privater Speicherbereich für Benutzerspezifische Anpassungen, Im- und Export-Möglichkeiten sowie Kollaborationsfunktionen sind aber nicht vorhanden. Hinsichtlich der Anwendung der Muster existiert eine Unterstützung der Benutzer in Form eines graphischen Editors, mit dessen Hilfe sich Benutzeroberflächen durch Komposition der Patterns entwerfen lassen. Zudem kann eine automatisierte Code-Generierung erfolgen. UPADE ist der Kategorie der Pattern-basierten Entwurfs-Werkzeuge zuzuordnen.

Das Pattern-Tool Damask verfügt über Authoring-, Browsing- und Suchfunktionen, während eine Unterstützung der Zusammenarbeit von Benutzern gänzlich fehlt. Die Patterns sind mit UI-Design-Fragmenten angereichert, die eine gleichzeitige Erstellung von Sketch-basierten UI-Entwürfen für unterschiedliche Endgeräte-Plattformen erlauben. Damask ist ein typischer Vertreter der Kategorie der Pattern-basierten Entwurfs-Werkzeuge.

Die Montreal Online Usability Patterns Digital Library (MOUDIL) war zunächst für das reine Management von Patterns konzipiert und verfügt daher über vielfältige diesbezügliche Funktionalitäten. Es fehlt lediglich ein privater Speicherbereich für persönliche Änderungen und Anpassungen. Es können Kommentare zu Mustern hinterlegt werden, die für die anderen User einsehbar sind. Das Werkzeug kontrolliert den vollständigen Lebenszyklus der Muster auf Basis der sogenannten Seven C's Methodology. In einer späteren Version unterstützt es seine Benutzer durch kontextabhängige Ratschläge hinsichtlich der Verwendung der Patterns und erlaubt die automatische Generierung von Java- oder C#-Klassen. Aus diesem Grund wird MOUDIL den Pattern-basierten Entwurfs-Werkzeugen zugeordnet.

Das Pattern-Tool Management of User Interface Patterns (MUIP) verfügt bis auf die fehlenden Im- bzw. Export-Möglichkeiten über alle untersuchten Management-Funktionen. Auch hier können Benutzer die Beschreibungen der Muster kommentieren. Eine besondere Rolle spielt das Beschreibungselement <Forces>, durch das das Auffinden und die Auswahl geeigneter Patterns vereinfacht werden soll. MUIP gehört zur Kategorie der Pattern-Management-Werkzeuge.

Der Fokus des Pattern Language Assistant (PLAss) liegt, wie der Name bereits andeutet, auf der Spezifikation von Pattern-Sprachen. Die Definition der betreffenden Patterns ist nicht an eine bestimmte Beschreibungssprache gebunden. Die Beschreibungselemente können vielmehr für jede Pattern-Language separat festgelegt bzw. ausgewählt werden. Das Werkzeug bietet zudem automatisierte Unterstützung bei der Spezifikation von Pattern-Hierarchien und -Relationen.

Im HCI Pattern Language Management Tool (HCI PLMT) stehen die Management-Funktionen Browsing, Searching, Modification und Relating zur Verfügung. Hinsichtlich der Suche werden Schlüsselwort- und Volltext-Suche angeboten. Es wird automatisch in den Beschreibungselementen nach Namen vorhandener Patterns gesucht und auf dieser Basis Vorschläge für mögliche Beziehungen zwischen Patterns generiert. HCI PLMT gehört zur Kategorie der Pattern-Management-Werkzeuge.

In der Quelle zum Werkzeug Patterns in Modelling (PIM) Tool finden sich keine Hinweise auf eventuell vorhandene Management- und Kollaborations-Funktionalitäten. Der Fokus liegt auf der Unterstützung der automatisierten Verwendung der Patterns, die in vier Prozess-Schritten vonstattengeht. Die Muster beinhalten potenziell Task-, Dialog-, Präsentations- und Layout- Fragmente, wobei ersteres aus statischen und variablen Anteilen bestehen

kann. Bei der Anwendung der Muster werden zunächst die eventuell vorhandenen variablen Teile interaktiv an den jeweiligen Nutzungskontext angepasst und anschließend in ein sogenanntes UI-Multi-Modell integriert. Die beschriebene Version unterstützt ausschließlich Task-Modell-Fragmente. Das PIM Tool ist ein Pattern-basiertes Entwurfs-Werkzeug.

Der Task Pattern Wizard bietet Browsing-Funktionalitäten und basiert auf denselben theoretischen Grundlagen wie das zuvor genannte PIM Tool. Auch hier besitzen die Patterns Task-, Dialog-, Präsentations- und Layout- Fragmente, die mithilfe von XUL (siehe Kapitel 2.4.1.2) formal beschrieben sind. Sie werden im Rahmen der Anwendung der Muster in übergeordnete UI-Modelle integriert. Demnach ist auch der Task Pattern Wizard ein Pattern-basiertes Entwurfs-Werkzeug.

Zu allen bisher beschriebenen Pattern-Werkzeugen gibt es laut der jeweiligen Quellen existierende Implementierungen, die jedoch allesamt nicht frei zugänglich sind. Die folgenden Tools besitzen Browser-basierte Front-Ends und können über das Internet genutzt werden.

Das Tool PatternWiki verfügt sowohl über ausgefeilte Management- als auch Kollaborations-Funktionen. Es fehlen lediglich der private Speicherbereich für Benutzer-spezifische Anpassungen und die Möglichkeiten zum Im- bzw. Export von Pattern-Beschreibungen. Bei der Anwendung der Muster können auf XSLT182 basierende Transformationen zur Generierung von UI-Code zum Einsatz kommen. PatternWiki weist die klassischen Merkmale eines Pattern-Management-Werkzeugs auf, wird aber wegen der vorhandenen Möglichkeiten zur Code-Generierung dennoch der Kategorie der Pattern-basierten Entwurfs-Werkzeuge zugeordnet.

Alle untersuchten Webseiten zu Pattern-Sammlungen bieten grundsätzlich Möglichkeiten zum Browsing und zur Suche von Mustern. Die Benutzer von Patterns in Interaction Design und Design of Sites können zusätzlich Pattern-Beschreibungen kommentieren während bei der Yahoo Design Pattern Library die Möglichkeit zur Teilnahme an Diskussions-Foren besteht. Der Quince UX Patterns Explorer erlaubt die Registrierung als dann namentlich bekannter User. Zudem werden von den Werkzeugen teilweise Hintergrundinformationen zu Patterns, Verweise auf andere Pattern-Sammlungen und Möglichkeiten zum Versenden von Nachrichten und zum Vorschlagen neuer Muster angeboten. Diese Tools gehören der Kategorie der Pattern-Katalog-Werkzeuge an. Einzige Ausnahme bildet hierbei der Quince UX Patterns Explorer, für den dies nur für die öffentliche Version ohne Benutzer-Registrierung zutrifft. Wenn sich der Benutzer jedoch registriert und anmeldet, so stehen ihm zusätzlich Pattern-Management-Funktionalitäten zur Verfügung. Bei der kostenpflichtigen Variante Quince Pro können ergänzend private Speicherbereiche und eine automatisierte Anwendung der Patterns für die UI-Modellierung genutzt werden. Diese Version ist also den Pattern-basierten Entwurfs-Werkzeugen zuzuordnen.

Die vollständige Beschreibung der Untersuchungsergebnisse in Tabellenform befindet sich in Anhang D.

[182] Siehe http://www.w3.org/TR/xslt

4. Entwurf der Entwicklungsumgebung

Die vorliegende Dissertation verfolgt die Grundidee der Kombination von sowohl im Software-Engineering als auch in der HCI-Gemeinschaft anerkannten und etablierten Techniken und Verfahren zur Entwicklung von Benutzeroberflächen in einem gemeinsamen Framework. Wie in Abbildung 12 auf abstrakter Ebene dargestellt, handelt es sich hierbei um das Zusammenspiel von modellgetriebener und musterbasierter Softwareentwicklung. Zudem besteht die Möglichkeit, Rückmeldungen zur Gebrauchstauglichkeit hinsichtlich der hierdurch erzielten Ergebnisse den verwendeten Modellen und/oder Mustern zuzuordnen und somit etwaigen Verbesserungsbedarf offenzulegen.

Bei modellgetriebenen Verfahren erfolgt die Entwicklung von Benutzeroberflächen bzw. deren Anpassung an neue Hard- und Software-Plattformen nicht durch direkte Erstellung oder Veränderung von Programm-Code. Es werden vielmehr die maßgeblichen Aspekte der Benutzerschnittstelle mithilfe diverser abstrakter Modelle beschrieben, die anschließend Schritt für Schritt durch möglichst automatische Modelltransformationen in das letztendliche User-Interface überführt werden (siehe Kapitel 2). Die Erstellung und Pflege dieser Modelle sowie die Sicherstellung der Konsistenz bei Erweiterung oder Veränderung ist mitunter komplex, aufwändig und fehleranfällig [197].

Bei der musterbasierten Software-Entwicklung wird im Sinne der Wiederverwendbarkeit Expertenwissen gesammelt, in Form von Mustern dokumentiert und somit für Andere verfügbar gemacht. Im Wesentlichen werden hierbei Lösungsansätze zu wiederkehrenden Problemstellungen für definierte Anwendungskontexte beschrieben (siehe Kapitel 3).

Die Idee ist nun, Muster auf eine geeignete Art und Weise so zu formalisieren und um weitere Informationen anzureichern, dass mit Ihrer Hilfe die beschriebenen Schwächen der modellgetriebenen Software-Entwicklung ausgeglichen oder zumindest gelindert werden. Vereinfacht kann gesagt werden, dass Muster dazu dienen, eine Wiederverwendbarkeit bei der Erstellung und Überführung von UI-Modellen zu realisieren.

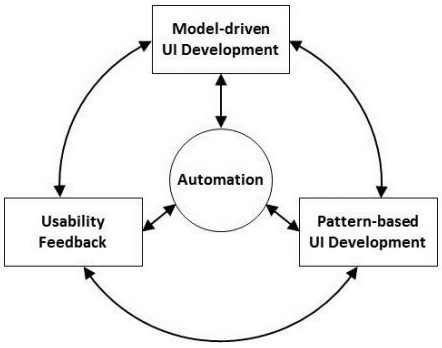


Abbildung 12 GRUNDIDEE DES PAMGIS FRAMEWORKS

Zu Beginn des Promotions-Projekts wurde der Gedanke verfolgt, bestehende modellgetriebene Entwicklungsumgebungen für Benutzerschnittstellen zu untersuchen und, basierend auf den entsprechenden Ergebnissen, ein geeignetes MB-UIDE auszuwählen, das dann anschließend um den oben genannten musterbasierten Anteil erweitert wird. Dieser Plan musste aber letztlich aufgegeben werden, da über keines der betrachteten Frameworks

vollständige und ausreichend detaillierte Informationen vorlagen. Deshalb wurden Fokus und Umfang der Dissertation entsprechend geändert und die durch die Recherchen gewonnenen Erkenntnisse dazu genutzt, neben dem Pattern-basierten auch den modellgetriebenen Teil der kombinierten UI-Entwicklungsumgebung neu zu entwerfen. Dafür wird die Integration des Usability-Feedback nicht im Detail ausgearbeitet, sondern nur skizziert. Das Ergebnis ist das im Folgenden beschriebene *Framework for Pattern-based Modeling and Generation of Interactive Systems* (PaMGIS).

Hierzu wird in Kapitel 4.1 zunächst zu einigen grundsätzlichen Design-Entscheidungen Stellung genommen. Kapitel 4.1.2 befasst sich mit der generellen Architektur des Frameworks während in Kapitel 4.3 das Rollenmodell von PaMGIS erläutert ist. Die zugrundeliegende Entwicklungsmethodik wird in Kapitel 4.4 beschrieben. Detailaspekte des modellgetriebenen Anteils der Entwicklungsumgebung werden in Kapitel 4.6 erklärt. Die musterbasierten Artefakte und Mechanismen sind schließlich in Kapitel 4.7 zusammengefasst.

4.1. Grundsätzliche Design-Entscheidungen

In diesem Kapitel werden grundsätzliche Design-Entscheidungen für PaMGIS hergeleitet und beschrieben.

4.1.1. Modellgetriebener Anteil von PaMGIS

Hinsichtlich des modellgetriebenen Teils von PaMGIS wurden im Rahmen der vorliegenden Dissertation neben den grundlegenden Aspekten (siehe Kapitel 2.1 bis 2.3) insgesamt acht Beschreibungssprachen (siehe Kapitel 2.4) und 18 Entwicklungsumgebungen (siehe Kapitel 2.5) im Detail untersucht.

Viele der recherchierten MB-UIDE gehen auf die Zeit vor der Einführung des *Cameleon Reference Framework* (CRF), das im Zeitraum von 2001 bis 2003 entwickelt wurde, zurück. Danach entstanden nur noch die UI-Entwicklungsumgebungen TERESA, SUPPLE bzw. SUPPLE++ und MARIAE.

Zusammenfassend lässt sich sagen, dass bei den MB-UIDE durchaus sehr unterschiedliche Modellierungsansätze verfolgt werden. So kommen beispielsweise explizite Datenmodelle, die häufig auch als Domänen-Modelle bezeichnet werden, bei UIDE, ITS, GENIUS, TRIDENT, AME, JANUS, FUSE, MECANO, TADEUS (Rostock), MOBI-D, TEALLACH, TADEUS (Linz), TERESA, SUPPLE/SUPPLE++ und MARIAE zum Einsatz. Aufgabenmodellierung in einem gesonderten Task-Modell erfolgt bei ADEPT, TRIDENT, MASTERMIND, FUSE, TADEUS (Rostock), MOBI-D, TEALLACH, TADEUS (Linz), TERESA und MARIAE. Die Berücksichtigung von Eigenschaften der Benutzer, die in einem separaten User-Modell definiert werden, erfolgt bei UIDE, ADEPT, TRIDENT, FUSE, TADEUS (Rostock), MOBI-D, TADEUS (Linz) und SUPPLE/SUPPLE++. Hingegen wird ein explizites Device-Modell zur Spezifikation von Eigenschaften der verwendeten Endgeräte nur bei SUPPLE/SUPPLE++ verwendet.

Aufgrund der durchgeführten Literatur-Recherche ist zu vermuten, dass eine Weiterentwicklung nur noch bei MARIAE stattfindet (siehe Abbildung 11 auf Seite 39). Eine vollständige Auflistung der Ergebnisse der Recherche befindet sich in Anhang B.

Aufgrund der Vielfalt der vorhandenen Modellierungsansätze und der Tatsache, dass für keines der untersuchten MB-UIDE vollständige und hinreichend detaillierte Informationen vorlagen, wurde keine der Entwicklungsumgebungen als direkte Basis für die Spezifikation von PaMGIS verwendet. Natürlich dienten die diesbezüglichen Recherche-Ergebnisse aber als wertvoller Input bei der Gestaltung von PaMGIS.

PaMGIS orientiert sich am *Cameleon Reference Framework*, das als umfassende, vereinheitlichende und in sich schlüssige Architektur sehr gut für diese Zwecke geeignet ist. Zum Einsatz kommen die ontologischen Domänen- und Benutzungskontext-Modelle[183] des CRF. Das Domänen-Modell besteht demnach aus einem Aufgaben-Modell (engl. Task Model) und einem Konzept-Modell (engl. Concept Model), das praktisch einem Datenmodell der Benutzeroberfläche entspricht. Die Modellierung des Benutzungskontextes erfolgt mithilfe eines Benutzer-Modells (engl. User Model) und eines Umgebungs-Modells (engl. Environment Model). Das ebenfalls zum Benutzungskontext gehörende Plattform-Modell des CRF wird bei PaMGIS jedoch in zwei separate Modelle, das Endgeräte-Modell (engl. Device Model) und das UI-Programmiersprachen-Modell (engl. UI Toolkit Model), aufgeteilt. Ersteres spezifiziert alle relevanten Merkmale des Benutzerendgeräts während letzteres die in der verwendeten Programmiersprache zur Verfügung stehenden UI-Elemente beschreibt. Diese Maßnahme hilft bei der Vermeidung von Redundanzen in Fällen, in denen dieselbe Programmierumgebung auf mehreren, signifikant unterschiedlichen Hardware-Plattformen zum Einsatz kommt, beispielsweise Android[184] auf Mobiltelefonen und Tablet-Computern oder Java[185] auf Desktop-Computern und eingebetteten Systemen (engl. Embedded Systems). Die dynamischen Aspekte der Benutzerschnittstelle werden anhand eines zusätzlichen Dialog-Modells beschrieben. Dieses basiert, wie auch beim Rostocker TADEUS-System, auf Dialog-Graphen.

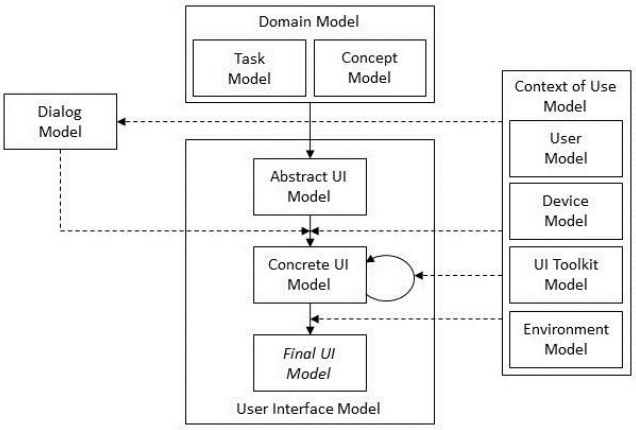


Abbildung 13 ÜBERSICHT DER VON IN PAMGIS VERWENDETEN MODELLE

Die Modellierung der eigentlichen Benutzerschnittstelle erfolgt, wie durch das CRF vorgegeben, durch die Ableitung eines abstrakten UI-Modells (AUI) aus den Domänen-Modellen, das dann mithilfe der Kontext-Modelle schrittweise über ein konkretes (CUI) in ein finales UI-

[183] Benutzungskontext-Modelle werden im weiteren Verlauf auch abkürzend als Kontext-Modell bezeichnet
[184] Betriebssystem für mobile Endgeräte der Firma Google
[185] Objektorientierte Programmiersprache der Firma Oracle

Modell (FUI) überführt wird. Ein grafischer Überblick über die in PaMGIS zum Einsatz kommenden Modelle und ihre Zusammenhänge wird in Abbildung 13 gezeigt.

Im Umfang der in dieser Dissertation vorgestellten Version von PaMGIS sind die Adaptions-Modelle des CRF, die zur dynamischen Anpassung der Benutzerschnittstelle zur Laufzeit dienen, nicht enthalten. Die gewählte Architektur läßt aber eine entsprechende Erweiterung zu einem späteren Zeitpunkt zu.

Ähnlich wie bei den UI-Entwicklungsumgebungen sind auch die untersuchten Beschreibungssprachen für die Benutzerschnittstellen relativ alt und werden größtenteils nicht mehr gepflegt (siehe Kapitel 2.4). Außerdem lagen auch hier meist nicht ausreichend detaillierte Informationen vor, um eine der Sprachen direkt zu übernehmen. Deshalb wurde der Entschluß gefasst, eine eigene, speziell an die Bedürfnisse von PaMGIS angepasste Beschreibungssprache für die Modelle zu entwickeln. Selbstverständlich sind dabei die Erkenntnisse aus den diesbezüglichen Recherchen eingeflossen. Die Spezifikation der Modelle erfolgt konform zur *Extensible Markup Language* (XML). Dies hat den Vorteil, dass dieses textbasierte Format sowohl für Menschen als auch für Maschinen lesbar ist. Zudem kann auch eine Vielzahl von bereits existierenden Werkzeugen, beispielsweise zur Transformation von entsprechend formulierten Inhalten, zur Anwendung kommen. Einzelheiten zur Spezifikation der PaMGIS-Modelle sind in Kapitel 4.6.2 beschrieben.

Eine weitere grundsätzliche Design-Entscheidung betrifft die Modellierungstiefe. Während beispielsweise bei ITS (siehe Kapitel 2.5.1.2) festgelegt werden kann bzw. muss, wie ein Objekt der Benutzerschnittstelle auf bestimmte Ereignisse (z. B. einfacher oder doppelter Maus-Klick) reagiert, so ist dies bei PaMGIS nicht notwendig. Hier endet die Modellierung auf der Ebene der durch die verwendete Programmiersprache bzw. das eingesetzte UI-Toolkit bereitgehaltenen UI-Objekte. Einerseits hat dies zur Folge, dass der Anwender des Frameworks nicht mehr die vollständige Freiheit hat, derartige Detailaspekte direkt zu beeinflussen, andererseits führt dies aber zu einer signifikanten Vereinfachung der Modellierung und des dafür erforderlichen Aufwandes. Zudem erscheint diese Limitierung aufgrund der Vielzahl von verfügbaren Programmiersprachen und Toolkits mehr als vertretbar.

4.1.2. Musterbasierter Anteil von PaMGIS

Bezüglich des musterbasierten Anteils von PaMGIS wurden im Rahmen der Dissertation neben den grundlegenden Aspekten (siehe Kapitel 3.1) insgesamt sechs formale Beschreibungssprachen für Muster (siehe Kapitel 3.2) und fünf existierende Pattern-Sammlungen (siehe Kapitel 3.3) untersucht. Zudem wurden vierzehn Pattern-Werkzeuge, darunter auch fünf Web-Seiten zu den vorher genannten Pattern-Sammlungen, analysiert und ausgewertet (siehe Kapitel 3.4).

Von den untersuchten Pattern-Beschreibungssprachen verfügen lediglich PLML 1.1 und PLML 1.2 jeweils mittels des Beschreibungselements <implementation> über die Möglichkeit Hinweise zur praktischen Realisierung der im Pattern angebotenen Lösung oder gegebenenfalls auch Code-Fragmente aufzunehmen. Während bei PLML 1.1 dieses Element gänzlich unstrukturiert ist, erfolgt im Fall von PLML 1.2 zwar eine Aufteilung in die drei Unterelemente <implementation-name>, <code> und <otherdetails>, die aber auch nicht weitergehend spezifiziert sind (siehe 3.2.1.2).

Direkte Unterstützung für einen modellgetriebenen Ansatz bietet einzig die Muster-Beschreibungssprache TPML, bei der das Element <Body> Teile eines Aufgaben-Modells aufnehmen kann. Dabei besteht die Möglichkeit, sowohl statische als auch variable Modellteile zu definieren. Letztere tragen dazu bei, abstraktere und vielseitiger einsetzbare Muster zu spezifizieren. Die variablen Modellteile werden bei der Anwendung des Patterns an den jeweils vorliegenden Benutzungskontext angepasst.

Bei den betrachteten Pattern-Sammlungen enthalten die Muster bei *Design of Sites*, der *Quince Pattern Library* und der *Yahoo Design Pattern Library* Informationen im Sinne des <implementation>-Elements von PLML 1.1 oder PLML 1.2. Bei *Design of Sites* und der *Quince Pattern Library* sind dies Hinweise zur praktischen Realisierung in Textform. In der *Yahoo Design Pattern Library* sind Code-Fragmente im Beschreibungselement <code-examples> hinterlegt.

Weder die untersuchten, formalen Beschreibungssprachen noch die Muster-Definitionen der genannten Pattern-Sammlungen verfügen über ausreichende Mittel, Muster mit Informationen zur Unterstützung einer modellgetriebenen Entwicklung von Benutzeroberflächen anzureichern.

Von allen dieser Hilfsmittel werden jeweils neue und interessante Aspekte eingebracht, die aber bei den anderen nicht oder nur rudimentär abgebildet werden. Es herrscht diesbezüglich also Mangel an Standardisierung. Aus diesem Grund wurde die Notwendigkeit gesehen, eine neue, vereinheitlichende Beschreibungssprache für Patterns zu entwickeln. Diese wird im Folgenden als *PaMGIS Pattern Specification Language* (PPSL) bezeichnet.

Mit PPSL sollen zwei Hauptziele erreicht werden. Einerseits ist dies die möglichst verlustfreie Überführung von bereits existierenden Mustern aus den betrachteten Pattern-Sammlungen bzw. von Patterns, die mithilfe der vorgestellten Beschreibungssprachen spezifiziert sind, in den PPSL-Formalismus. Andererseits sollen die Pattern-Definitionen alle für die automatische Verarbeitung in einem modellgetriebenen Ansatz notwendigen Informationen aufnehmen können. Um eine gleichzeitige Lesbarkeit für Mensch und Maschine zu erreichen, wird eine XML-konforme Notation eingesetzt.

Eine schematische Darstellung der Bedeutung von PPSL und deren Einbettung in das PaMGIS-Framework wird in Abbildung 14 gezeigt.

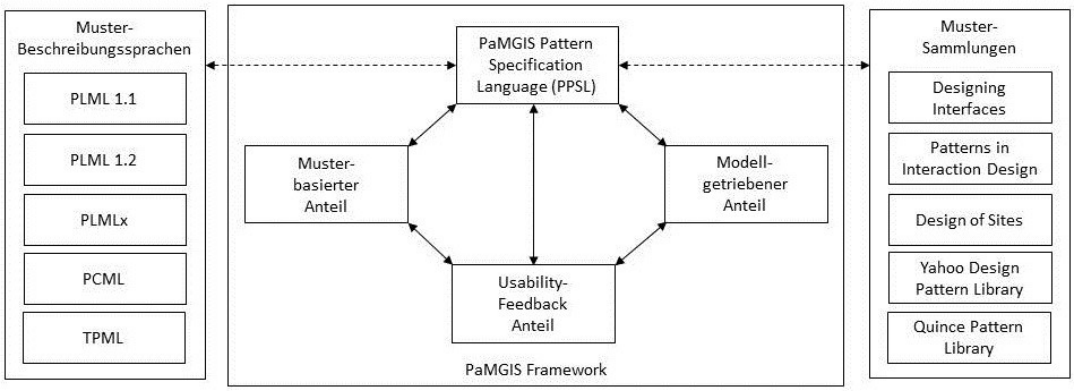


Abbildung 14 ÜBERSICHT ÜBER DIE ZUSAMMENHÄNGE ZWISCHEN DEN PAMGIS-KOMPONENTEN UND PPSL

Die betrachteten Pattern-Tools wurden hinsichtlich der von ihnen angebotenen Funktionalitäten untersucht. Für die Konzeption von PaMGIS sind hierbei besonders die Pattern-

basierten Entwurfs-Werkzeuge (siehe Definition in Kapitel 3.4.1) von Interesse. In diese Kategorie fallen UPADE, Damask, MOUDIL, PIM Tool, Task Pattern Wizard, Pattern Wiki und der Quince UX Patterns Explorer. Hierbei unterstützen Moudil, Task Pattern Wizard und Quince UX Patterns Explorer proprietäre und in der Literatur nicht vollständig beschriebene Beschreibungssprachen für Muster. Damask und Pattern Wiki basieren auf PLML, während UPADE PCML und PIM Tool TPML zur Grundlage haben.

Da der Pattern-basierte Teil von PaMGIS, anders als die genannten Tools, PPSL als zentrales Element aufweist, müssen entsprechende, unterstützende Werkzeuge entworfen werden. Hierbei spielen die Ergebnisse der Untersuchung der existierenden Pattern-Tools eine wichtige Rolle und dienen als wertvoller Input hinsichtlich der benötigten Funktionalitäten zur Verwaltung der Patterns und für die Zusammenarbeit von den Benutzern des Frameworks. Eine vollständige Auflistung der Ergebnisse der diesbezüglichen Recherche befindet sich in Anhang D.

4.2. Genereller Aufbau

Der grundlegende Aufbau von PaMGIS wird in Abbildung 15 dargestellt.

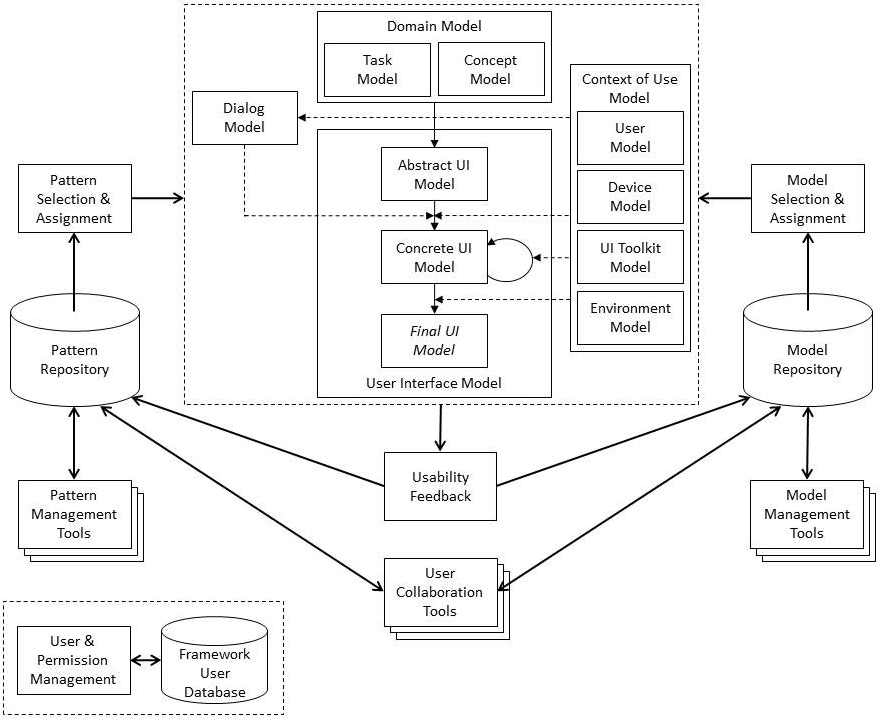


Abbildung 15 LOGISCHER AUFBAU DES PAMGIS FRAMEWORKS

Der modellgetriebene Teil von PaMGIS besteht aus den bereits in Kapitel 4.1.1 aufgezeigten Modellen und entsprechenden Transformationen zur Erzeugung der UI-Modelle auf den verschiedenen Abstraktionsebenen. Ergänzt wird dies, wie auf der rechten Seite von Abbildung 15 gezeigt, durch ein Modell-Repository und Werkzeuge zur Erstellung, Pflege und Auswahl der Modelle. Einerseits beherbergt das Modell-Repository die Metamodelle der zur Anwendung kommenden Modelltypen, die analog zum Sprachgebrauch des CRF als

ontologische Modelle bezeichnet werden. Andererseits nimmt es auch deren Instanzen, also die eigentlich zur Verfügung stehenden Modelle, auf. Hierbei wird zwischen den archetypischen und den benutzerdefinierten Modellen unterschieden. Archetypische Modelle sind Instanzen der Metamodelle, die fest zum PaMGIS-Framework gehören und nicht durch einen normalen Benutzer geändert werden dürfen. Diese erhalten stets Kopien der von ihnen ausgewählten, archetypischen Modelle und können diese in einem für sie reservierten Arbeitsbereich (engl. User Workspace) verändern und benützen. Eine detaillierte Beschreibung des modellgetriebenen Anteils von PaMGIS befindet sich in Kapitel 4.6.

Entsprechender Weise verfügt der musterbasierte Teil über ein Pattern-Repository und Werkzeuge zur Verwaltung und Auswahl der Muster (siehe Abbildung 15 linke Seite). Im Pattern-Repository werden sowohl das Metamodell der Muster, d.h. die Spezifikation der Pattern-Beschreibungssprache PPSL, als auch deren Instanzen, also die eigentlich zur Verfügung stehenden Muster, gespeichert. Zudem besteht die Möglichkeit, dort auf Basis der enthaltenen Muster spezifizierte Pattern-Sprachen zu hinterlegen. Analog zu den Modellen wird auch hier zwischen archetypischen und benutzerdefinierten Patterns bzw. Pattern-Sprachen unterschieden. Letzere finden bei der Erstellung der Modelle Verwendung. Die Beschreibung des musterbasierten Anteils befindet sich in Kapitel 4.7.

Als Unterstützung der PaMGIS-Benutzer bei der Erstellung und Pflege sowohl der Modelle als auch der Patterns stehen entsprechende Werkzeuge für deren Zusammenarbeit zur Verfügung (engl. User Colloboration Tools).

Die Komponente zum Usability-Feedback ermöglicht, entsprechende Erkenntnisse, die beispielsweise durch Usability-Evaluation der erzeugten Benutzeroberfläche gewonnen wurden, bei den betreffenden Patterns- und Modell-Spezifikationen zu hinterlegen. Auf diese Weise können diese Informationen als wertvoller Input bei deren Weiterentwicklung dienen.

Letztlich verfügt das Framework noch über eine Benutzerverwaltung, die aus einer Benutzer-Datenbank und einem zugehörigen Werkzeug für die Erstellung und Pflege von Benutzern, Rollen und Berechtigungen besteht. Das Rollenkonzept von PaMGIS wird in Kapitel 4.3 erläutert.

4.3. Rollen-Modell

Das PaMGIS-Framwork verfügt über eine Benutzerverwaltung, die eine Zuweisung von Berechtigungen an die jeweiligen Benutzer mittels eines Rollenmodells erlaubt. Folgende Rollen stehen hierfür zur Verfügung:

1. Framework-Administrator
2. Pattern-Administrator
3. Modell-Administrator
4. Anonymer Benutzer
5. Registrierter Benutzer
6. Power-User

Ein Framework-Administrator hat grundsätzlich Vollzugriff auf alle von PaMGIS angebotenen Funktionen. Insbesondere ist einzig diese Rolle dazu berechtigt, die ontologischen Modelle sowohl des modellgetriebenen, als auch des musterbasierten Framework-Anteils zu verän-

dern. Darüber hinaus verantwortet der Framework-Administrator die Benutzerverwaltung, indem er Benutzer anlegt und diese durch Zuweisung von Rollen entsprechende Berechtigungen vergibt.

Ein Pattern-Administrator übernimmt die Aufgabe der Erstellung und Pflege der archetypischen Muster und Pattern-Sprachen. Sobald er diese freigibt, stehen sie für die verschiedenen Benutzer-Rollen zur Verfügung. Die Spezifikation von PPSL darf von ihm nur gelesen, jedoch nicht geändert werden.

Analog dazu erstellt und pflegt der Modell-Administrator die archetypischen Modelle. Auch hier wird durch die Freigabe dafür gesorgt, dass Benutzer diese verwenden können. Bezüglich der ontologischen Modelle besitzt diese Rolle nur Lese-Rechte.

Anonyme Benutzer haben eingeschränkten, lesenden Zugriff auf archetypische Muster. Die Einschränkung bezieht sich einerseits darauf, dass hierbei nur eine eigens dafür freigegebene Auswahl von Patterns zur Verfügung steht und andererseits nicht alle PPSL-Beschreibungselemente für diese Art der Benutzer sichtbar sind. Ähnlich wie bei der Web-Seite zur Pattern-Sammlung *Designing Interfaces* (siehe Kapitel 3.4.2.10) stehen lediglich Funktionalitäten zum Browsen und Suchen von Mustern zur Verfügung. Zugriffe auf den modellgetriebenen Anteil von PaMGIS sind nicht erlaubt.

Ein registrierter Benutzer verfügt über einen Eintrag in der Benutzer-Datenbank und hat gegenüber anonymen Benutzern deutlich erweiterte Rechte. Er kann grundsätzlich alle freigegebenen, archetypischen Muster und Pattern-Sprachen einsehen. Allerdings besteht auch hier die Einschränkung, dass ihm nicht alle Beschreibungselemente von PPSL zugänglich sind. Insbesondere besteht kein Zugriff auf die für den modellgetriebenen Teil von PaMGIS benötigten Informationen. Registrierten Benutzern steht ein persönlicher Arbeitsbereich im Pattern-Repository zur Verfügung. In diesem können neue Patterns angelegt oder Kopien von archetypischen Mustern oder ganzer Pattern-Sprachen abgelegt und anschließend unter Zuhilfenahme der Pattern-Management-Tools nach Belieben erweitert und/oder verändert werden. Zudem können die Kollaborationswerkzeuge für den Pattern-basierten Teil von PaMGIS genutzt werden. Beispielsweise ist es möglich ein neues Pattern oder eine Weiterentwicklung eines bestehenden Patterns für die Aufnahme in die Menge der archetypischen Muster vorzuschlagen.

Ein Power-User ist ebenfalls in der Benutzerverwaltung registriert und verfügt über Berechtigungen sowohl für den musterbasierten als auch für den modellgetriebenen Teil von PaMGIS. Auch ihm steht ein User-Workspace zur Verfügung, den er in projektspezifische Unterbereiche, die sogenannten PaMGIS-Projekte, strukturieren kann. Dort können Patterns und Modelle neu angelegt werden oder jeweils Kopien von archetypischen Mustern, Pattern-Sprachen und/oder Modellen verwendet und weiterentwickelt werden. Hierfür stehen die Funktionalitäten der Pattern-Management-, Modell-Management- und Kollaborations-Tools zur Verfügung. Ferner können Power-User die Muster und Modelle dazu nutzen, um Benutzeroberflächen zu generieren. Außerdem haben sie die Möglichkeit, Usability-Feedback an die betreffenden Muster- und Modell-Spezifikationen zurückzuführen.

Zusammenfassend lässt sich sagen, dass die Berechtigungen für anonyme und registrierte Benutzer auf den Pattern-basierten Anteil von PaMGIS beschränkt sind. Der Framework-Administrator verantwortet neben der Benutzerverwaltung hauptsächlich die Metamodelle, d.h. die ontologischen Modelle von PaMGIS während sich die Pattern- und Modell-

Administratoren um die archetypischen Muster und Pattern-Sprachen bzw. die archetypischen Modelle kümmern. Die Power-User können in ihrem persönlichen User-Work-Space PaMGIS-Projekte anlegen und dort entweder mit Kopien der von ihnen ausgewählten archetypischen Mustern, Pattern-Sprachen und Modellen arbeiten oder auch jeweils neue auf Basis der verfügbaren Metamodelle erzeugen. Diese kann er dann zur Generierung von User-Interfaces nutzen.

4.4. Entwicklungsmethodik

Im vorliegenden Kapitel werden die wichtigsten Prozesse zur Entwicklung von Mustern bzw. Pattern-Sprachen, Modellen und der Generierung von Benutzeroberflächen beschrieben.

4.4.1. Erstellung von Mustern und Pattern-Sprachen

Wie bereits erwähnt, wird zwischen den ontologischen Modellen (Metamodelle), den archetypischen Patterns bzw. Pattern-Sprachen und den benutzerdefinierten Patterns bzw. Pattern-Sprachen unterschieden. Im Pattern-Repository sind hierfür unterschiedliche Bereiche reserviert. Eine Pattern-Sprache bezeichnet im Sinne der Definition in Kapitel 3.3.1.2 eine Menge von zueinander in Beziehung stehenden Mustern für eine gegebene Anwendungsdomäne. Der besseren Lesbarkeit halber beziehen sich die folgenden Beschreibungen jeweils auf einzelne Muster, gelten jedoch analog auch für ganze Pattern-Sprachen.

Der Framework-Administrator erstellt und pflegt die ontologischen Modelle der Muster, d.h. die Spezifikationen der zur Verfügung stehenden PPSL-Versionen. Sobald eine Version freigegeben ist, können Pattern-Administratoren, registrierte Benutzer und Power-User auf deren Basis durch Instanziierung neue Patterns erzeugen und spezifizieren. Im Falle des Pattern-Administrators bedeutet das, dass er dadurch neue archetypische Muster zum Pattern-Repository hinzufügen kann. Analog dazu können registrierte Benutzer und Power-User neue Muster in ihrem jeweiligen User-Work-Space (UWS) anlegen.

Wollen registrierte Benutzer und Power-User archetypische Patterns verwenden, so müssen sie dazu eine Kopie des betreffenden Musters in ihrem persönlichen Arbeitsbereich anlegen. Das bedeutet, dass sie niemals direkt mit archetypischen Patterns, sondern stets mit entsprechenden Kopien arbeiten.

Es besteht auch die Möglichkeit, Kopien von bestehenden Patterns zu erzeugen und diese anschließend zu verändern. Auf diese Weise können neue Versionen bzw. Revisionen von Mustern entstehen. In der Regel führen hierbei größere, signifikante Änderungen zu einer neuen Version während kleinere Anpassungen als neue Revisionen eines Patterns bezeichnet werden (siehe hierzu auch die Spezifikation von PPSL in Kapitel 4.7.2). Ein durch den Pattern-Administrator geändertes und freigegebenes Muster steht den Benutzern automatisch als archetypisches Pattern zur Verfügung. Kopien, die durch registrierte Benutzer oder Power-User erzeugt und abgeändert wurden, können diese entweder für ihre Zwecke nutzen oder auch zur Aufnahme in die Liste der archetypischen Patterns vorgeschlagen. Eine Entscheidung darüber trifft dann der Pattern-Administrator.

Zieht ein Pattern-Administrator ein archetypisches Pattern zurück, so bleibt dieses zwar aus Konsistenzgründen erhalten, steht aber für eine neuerliche Verwendung nicht mehr zur

Verfügung. Gleiches gilt für die Rückziehung von PPSL-Versionen durch den Framework-Administrator.

Eine Übersicht über die wichtigsten Prozesse bezüglich der Erstellung von Patterns und den jeweils berechtigten Rollen wird in Tabelle 16 dargestellt.

Tabelle 16 ÜBERSICHT ÜBER DIE WICHTIGSTEN PROZESSE ZUR ERSTELLUNG VON PATTERNS

Prozesse zur Pattern-Erstellung	Rollen					
	Framework-Administrator	Pattern-Administrator	Modell-Administrator	Anonymer Benutzer	Registrierter Benutzer	Power-User
Änderung Metamodelle	X					
Neuanlage eines archetypisches Patterns	X[a)]	X				
Neue Version eines archetypisches Patterns	X[a)]	X				
Neuanlage eines benutzerdefinierten Patterns	X[b)]	X[b)]			X[c)]	X[c)]
Kopieren eines archetypischen Patterns in UWS	X[b)]	X[b)]			X[c)]	X[c)]
Neue Version eines benutzerdefinierten Patterns	X[b)]	X[b)]			X[c)]	X[c)]
Vorschlagen eines neuen archetypischen Patterns	X[b)]	X[b)]			X[c)]	X[c)]
Entscheidung über eingereichten Vorschlag	X[a)]	X				

Legende:

a) Zu Support-Zwecken mit Einwilligung des Pattern-Administrators
b) Zu Support-Zwecken mit Einwilligung des betreffenden Benutzers
c) Nur im jeweils zugewiesenen persönlichen Arbeitsbereich (UWS)

In Tabelle 17 sind die Berechtigungen der Rollen hinsichtlich der verschiedenen Bereiche des Pattern-Repository zusammengefasst.

Tabelle 17 ÜBERSICHT ÜBER DIE BERECHTIGUNGEN DER ROLLEN FÜR DAS PATTERN-REPOSITORY

Pattern-Repository	Rollen					
	Framework-Administrator	Pattern-Administrator	Modell-Administrator	Anonymer Benutzer	Registrierter Benutzer	Power-User
Ontologischer Bereich	r/w	r			r[d)]	r
Archetypischer Bereich	r/w[a)]	r/w		r[c)]	r[e)]	r[g)]
User-Work-Space	r/w[b)]	r/w[b)]			r/w[f)]	r/w[f)]

Legende:

r Lesender Zugriff (engl. read)
w Schreibender Zugriff (engl. write)
a) Zu Support-Zwecken mit Einwilligung des Pattern-Administrators
b) Zu Support-Zwecken mit Einwilligung des betreffenden Benutzers
c) Nur eigens dafür freigegebene archetypischen Patterns, nur wenige Beschreibungselemente
d) Nur aktuelle Version von PPSL
e) Alle freigegebenen archetypischen Patterns, nicht die für den modellgetriebenen Teil von PaMGIS reservierten Beschreibungselemente (siehe Element <PaMGIS> in Kapitel 4.7.2)
f) Nur für den jeweils zugewiesenen persönlichen Arbeitsbereich
g) Alle freigegeben archetypischen Patterns

4.4.2. Erstellung von Modellen

Dieses Kapitel befasst sich mit den Prozessen zur Erstellung derjenigen Modellen, die praktisch als Input für das Generierungsverfahren für die Benutzeroberflächen dienen. Dazu zählen einerseits die Domänen- und Kontext-Modelle und andererseits auch die Dialog-Modelle, die von diesen abgeleitet werden. Die Prozesse bezüglich der eigentlichen UI-Modelle auf den verschiedenen Abstraktionsebenen in Form von AUI, CUI und FUI werden in Kapitel 4.4.3 behandelt.

Die Erstellung der genannten Modelle erfolgt in analoger Weise zu den Verfahren für die Erstellung von Patterns und Pattern-Sprachen. Zudem ist auch das Modell-Repository, ähnlich wie schon das Pattern-Repository, in Bereiche für ontologische, archetypische und benutzerdefinierte Modelle eingeteilt.

Der Rolle des Framework-Administrators ist die Pflege der ontologischen, also der Metamodelle, vorbehalten. Durch Freigabe eines Metamodells steht dieses dem Modell-Administrator und den Power-Usern zur Verfügung. Ein Modell-Administrator kann auf deren Basis durch Instanziierung neue archetypische Modelle erzeugen und spezifizieren. Power-User

können gleichermaßen neue Modelle in dem ihnen zugeordneten UWS anlegen und natürlich auch die angebotenen archetypischen Modelle verwenden. Wie bei den Patterns wird hierbei eine Kopie im persönlichen Arbeitsbereich angelegt. Darüber hinaus können auch hier der sowohl Modell-Administrator als auch die Power-User Kopien beliebiger Modelle erzeugen und nach ihren Wünschen verändern. Somit besteht die Möglichkeit, neue Versionen bzw. Revisionen von archetypischen bzw. benutzerdefinierten Modellen anzulegen. Power-User können diese dann zur Aufnahme in die Menge der archetypischen Modelle vorschlagen. Die Entscheidung darüber obliegt dem Modell-Administrator. Wird eine Version eines archetypischen Modells durch einen Modell-Administrator zurückgezogen, so bleibt dieses zwar zur Erhaltung der Konsistenz bestehen, steht aber für eine neuerliche Benutzung nicht mehr zur Verfügung.

Die anderen Rollen des Frameworks, also der Pattern-Administrator sowie die anonymen und registrierten Benutzer, haben keinerlei Berechtigungen hinsichtlich des modellgetriebenen Anteils von PaMGIS.

Eine Übersicht über die wichtigsten Prozesse bezüglich der Erstellung von Modellen und den jeweils berechtigten Rollen wird in Tabelle 18 dargestellt.

Tabelle 18 ÜBERSICHT ÜBER DIE WICHTIGSTEN PROZESSE ZUR ERSTELLUNG VON MODELLEN

Prozesse zur Modell-Erstellung	Rollen					
	Framework-Administrator	Pattern-Administrator	Modell-Administrator	Anonymer Benutzer	Registrierter Benutzer	Power-User
Änderung Metamodelle	X					
Neuanlage eines archetypischen Modells	X[a)]		X			
Neue Version eines archetypisches Modells	X[a)]		X			
Neuanlage eines benutzerdefinierten Modells	X[b)]		X[b)]			X[c)]
Kopieren eines archetypischen Modells in UWS	X[b)]		X[b)]			X[c)]
Neue Version eines benutzerdefinierten Modells	X[b)]		X[b)]			X[c)]
Vorschlagen eines neuen archetypischen Modells	X[b)]		X[b)]			X[c)]
Entscheidung über eigereichten Vorschlag	X[a)]		X			

Legende:

a) Zu Support-Zwecken mit Einwilligung des Pattern-Administrators
b) Zu Support-Zwecken mit Einwilligung des betreffenden Benutzers
c) Nur im jeweils zugewiesenen persönlichen Arbeitsbereich (UWS)

In Tabelle 19 sind die Berechtigungen der Rollen hinsichtlich der verschiedenen Bereiche des Modell-Repository zusammengefasst.

Tabelle 19 ÜBERSICHT ÜBER DIE BERECHTIGUNGEN DER ROLLEN FÜR DAS MODELL-REPOSITORY

Modell-Repository	Rollen					
	Framework-Administrator	Pattern-Administrator	Modell-Administrator	Anonymer Benutzer	Registrierter Benutzer	Power-User
Ontologischer Bereich	r/w		r			r
Archetypischer Bereich	r/w[a)]		r/w			r[c)]
User-Work-Space	r/w[b)]		r/w[b)]			r/w[d)]

Legende:

r Lesender Zugriff (engl. read)
w Schreibender Zugriff (engl. write)
a) Zu Support-Zwecken mit Einwilligung des Pattern-Administrators
b) Zu Support-Zwecken mit Einwilligung des betreffenden Benutzers
c) Alle freigegeben archetypischen Modelle
d) Nur für den jeweils zugewiesenen persönlichen Arbeitsbereich

4.4.3. Generierung von Benutzeroberflächen

Bei PaMGIS werden grundsätzlich folgende sechs Abstraktionsebenen unterschieden:

1. Domänen-Ebene (engl. Domain Level)
2. Benutzungskontext-Ebene (engl. Context of Use Level)
3. AUI-Ebene (engl. AUI Level)
4. CUI-Ebene (engl. CUI Level)
5. FUI-Ebene (engl. FUI Level)
6. Laufzeit-Ebene (engl. Runtime Level)

Wie in Abbildung 16 dargestellt, befindet sich auf der Domänen-Ebene das Domänen-Modell, das sich aus einem Aufgaben- und einem Konzept-Modell zusammensetzt. Die Benutzungskontext-Ebene, die abkürzend auch als Kontext-Ebene bezeichnet wird, umfasst das Kontext-Modell, das seinerseits aus den Teilmodellen für die Spezifikation des Benutzers, des zum Einsatz kommenden Endgerätes, des verwendeten UI-Toolkits und der zu berücksichtigenden Umgebungseinflüsse besteht. Zudem wird auch das Dialog-Modell, welches die dynamischen Aspekte der späteren Benutzeroberfläche definiert, dieser Ebene zugerechnet. Wie die Bezeichnung schon aussagt, befinden sich auf den AUI-, CUI- und FUI-Ebenen die jeweils entsprechenden UI-Modelle. Die Laufzeit-Ebene ist streng genommen nicht Teil des eigentlichen Frameworks. Sie fasst die für die Ausführung der erzeugten Benutzeroberflä-

chen benötigen Werkzeuge der Zielplattform wie Compiler und Interpreter zusammen. In Tabelle 16 wird auf dieser Ebene auch eine Laufzeit-Umgebung (engl. Runtime Environment) gezeigt, die in der vorliegenden Dissertation jedoch nicht näher beleuchtet wird. Solch eine Komponente würde dazu dienen, die UI-Modelle der unterschiedlichen Abstraktionsebenen direkt ausführbar zu machen und wird derzeit als Weiterentwicklungsmöglichkeit für PaMGIS angesehen (siehe hierzu auch Kapitel 6.3).

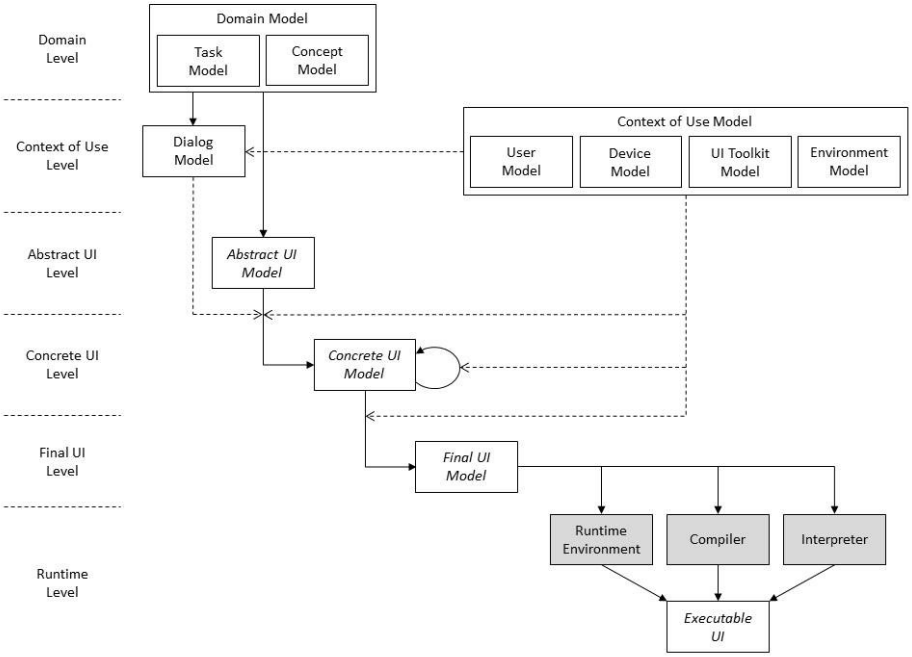


Abbildung 16 ÜBERSICHT ÜBER DIE ABSTRAKTIONSEBENEN VON PAMGIS

Der eigentliche Entwicklungs- und Generierungsprozess wird durch die Rolle des Power-Users durchgeführt. Der generelle Ablauf wird in Abbildung 17 dargestellt.

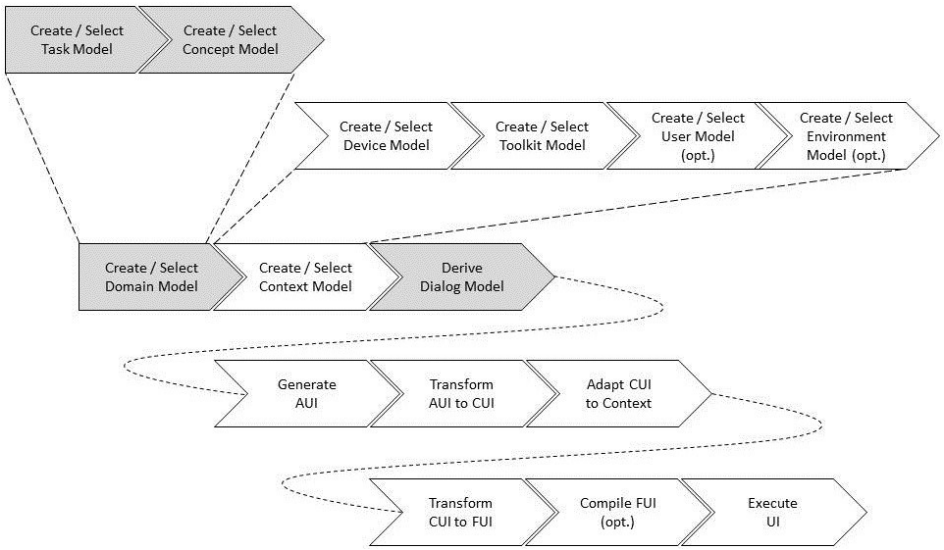


Abbildung 17 ABLAUF DER MODELLGETRIEBENEN ENTWICKLUNG VON BENUTZEROBERFLÄCHEN

Die in der Abbildung dunkel hinterlegten Prozessschritte können durch die Verwendung von Mustern aus dem Pattern-Repository unterstützt werden.

Das Domänen-Modell bildet den Ausgangspunkt für die Erstellung des abstrakten UI-Modells und somit für die schrittweise Generierung der Benutzeroberfläche. Aus diesem Grund ist dessen Erstellung der erste Schritt des PaMGIS-Entwicklungsprozesses. Der Power-User legt zunächst in seinem persönlichen Arbeitsbereich im Modell-Repository ein Projekt an. Alle zu diesem Projekt gehörenden Modelle werden dort gespeichert. Wie für alle Modelle gibt es hinsichtlich der Erstellung des Domänenmodells, das aus einem Paar von Aufgaben- und Konzept-Modell besteht, mehrere Möglichkeiten (siehe Kapitel 4.4.2): die Neuerstellung, die Nutzung von archetypischen Modellen oder die Änderung bzw. Erweiterung von bereits bestehenden Modellen. In allen Fällen wird der Benutzer durch einen entsprechenden Modell-Editor unterstützt. Dieser hilft auch bei der Spezifikation der wechselseitigen Beziehungen zwischen den beiden Teilmodellen. Zudem können Muster aus dem Pattern-Repository ausgewählt und angewendet werden. Das heißt, dass die in den betreffenden Pattern-Spezifikationen hinterlegten Modell-Fragmente, quasi wie Bausteine, automatisch an eine zuvor festgelegte Stelle in den Teilmodellen eingefügt werden.

Anschließend oder auch parallel dazu wird das Kontext-Modell entwickelt. Auch hierbei können wiederum archetypische Modelle verwendet werden. Sollten Änderungen an diesen Modellen notwendig sein, so steht wiederum ein Modell-Editor zur Verfügung. Eine Unterstützung durch Patterns ist hierbei aber nicht sinnvoll möglich.

Die dynamischen Aspekte der zukünftigen Benutzeroberfläche werden in Form von Dialogen und deren Ablaufreihenfolge in einem Dialog-Modell festgelegt. Es basiert, wie bei TADEUS (Rostock), auf Dialog-Graphen (siehe Kapitel 2.5.1.12). Auf Basis des Task-Modells wird hierbei ein erster Entwurf des Dialog-Modells unter Berücksichtigung des Kontext-Modells automatisch generiert. Dabei werden die Beziehungen zwischen den Aufgaben im Task-Modell ausgewertet und entsprechende Vorschläge für Zusammenfassungen von Aufgaben in Dialogen berechnet. Auch hier steht ein entsprechender Modell-Editor zur Verfügung. Für diejenigen Anteile des Aufgaben-Modells, die aus Patterns stammen, können auch entsprechende, ebenfalls im betreffenden Muster hinterlegte, Dialog-Modell-Fragmente bei der Konstruktion des Modells herangezogen werden.

Aus den im Domänen-Modell enthaltenen Informationen wird das abstrakte UI-Modell (AUI) generiert. Es umfasst hauptsächlich die Spezifikationen der zum Einsatz kommenden abstrakten User-Interface-Objekte (siehe Kapitel 4.6.2.1). Es handelt sich um eine vom Benutzungskontext unabhängige Darstellung der zukünftigen Benutzeroberfläche, d.h. es sind noch keinerlei Anforderungen und Spezifika bezüglich Endanwender, Endgerät, Programmierplattform und Umgebungseinflüssen berücksichtigt.

Das AUI-Modell wird nach dessen Fertigstellung in das konkrete UI-Modell (CUI) transformiert. Hierbei fließen einerseits die Informationen aus dem Kontext-Modell und andererseits die Struktur des Dialog-Modells mit ein. So werden beispielsweise die abstrakten UI-Objekte derjenigen Aufgaben eliminiert, die für den spezifizierten User oder auf dem zum Einsatz kommenden Endgerät generell nicht zur Verfügung stehen sollen. Die verbleibenden abstrakten UI-Objekte werden zudem unter Berücksichtigung des UI-Toolkit-Modells durch konkrete UI-Objekte ersetzt. Darüber hinaus findet noch eine grobe Anordnung der CUI-Objekte statt, d.h. die Lage der Objekte zueinander wird bestimmt. Zu einem späteren Zeitpunkt kann in Erwägung gezogen werden, hierfür ein Layout-Modell

einzuführen (siehe Kapitel 6.3). In der hier beschriebenen Version von PaMGIS ist dieses aber nicht enthalten.

Anschließend wird aus dem CUI-Modell automatisiert das finale UI-Modell (FUI) generiert. Hierbei werden die konkreten UI-Objekte durch die im UI-Toolkit tatsächlich verfügbaren Interaktionsobjekte ersetzt. Gegebenenfalls finden noch Anpassungen bezüglich des äußeren Erscheinungsbildes statt, wie beispielsweise die Änderung von Farben, Schriftart, Schriftgrad und/oder Größe der UI-Objekte.

Je nach Zielsprache muss das FUI noch kompiliert werden oder es wird direkt durch einen entsprechenden Interpreter ausgeführt. Eine weitere Möglichkeit wäre die direkte Ausführung der verschiedenen UI-Modelle mittels einer selbstentwickelten Laufzeitumgebung, die aber nicht im aktuellen Umfang des Frameworks geplant ist (siehe Kapitel 6.3).

4.4.4. Praktisches Anwendungsbeispiel

In diesem Unterkapitel wird die praktische Anwendung des PAMGIS-Frameworks aufgezeigt. Am Beispiel des „Poll“-Patterns aus der Pattern-Sammlung *Patterns in Interaction Design* von Martijn van Welie [183] werden die relevanten Modell-Fragmente abgeleitet und mittels PPSL beschrieben. Dann wird demonstriert, wie diese Fragmente dazu genutzt werden können, die Modelle der Benutzeroberfläche der Zielanwendung zu erstellen. Schließlich folgt die Erklärung, wie aus den Aufgaben-, Konzept- und Dialog-Modellen schrittweise eine Benutzerschnittstelle entsteht. Teile des Beispiels sind in englischer Sprache gehalten, da es auch im Rahmen einer internationalen Publikation verwendet wurde [200].

4.4.4.1. Das „Poll“-Pattern

Das „Poll“-Pattern dient gemäß der in [183] zu findenden Definition der Abfrage von Benutzermeinungen zu einem einzelnen, auf einer vorliegenden Webseite vorliegenden, Sachverhalt. Die originale Spezifikation des Musters ist in Tabelle 20 zusammengefasst.

Tabelle 20 AUSZUG DER ORIGINALEN BESCHREIBUNG DES „POLL“-PATTERNS VON MARTIJN VAN WELIE [183]

Element	Beschreibung
Problem	Users want to state their opinion about a certain statement that is relvant to the site’s content.
Solution	List the statements as exclusive options and present the results directly after voting.
Use when	You are designing a site where interaction with the users is desired. Typically this will be a News Site or Community Site where visitors are to be encouraged to share their opinions and improve interactivity.
How	The poll consists of two steps. First the list of options is presented, usually using radiobuttons, together with a 'vote' button. After clicking the vote button, the results are displayed. The results include both a percentage and an absolute number.
Why	A poll is a very simple and direct page element that invites users to in-teract with the site. Users can even do it anonymously so there is no barrier at all to participate. Polls are often linked to content on the site such as articles or products, and the results of a poll can be linked to a discussion in a Forum.

Es werden zunächst einige vorgegebene Fakten bzw. Aussagen zur Webseite angezeigt. Der Benutzer kann dann die Aussage, die seine Meinung am besten widerspiegelt, markieren

und seine Wahl absenden. Abschließend wird das Gesamtergebnis aller bisher eingegangenen Meinungen angezeigt.

Auf dieser Grundlage wurde das Muster zunächst dahingehend generalisiert, dass es die Abfrage jeglichen Inhalts erlaubt und sich nicht mehr nur auf Aspekte einer Webseite bezieht. Zudem wird das konkrete Aussehen der User-Interface-Elemente nicht wie bei van Welie vorgegeben, sondern es werden abstrakte UI-Elemente verwendet, die erst später schrittweise durch konkrete Elemente ersetzt werden. Der Ablauf der Befragung wird so gestaltet, dass sichergestellt ist, dass die Ergebnisse erst dann angezeigt werden, wenn die Teilnahme an der Umfrage tatsächlich erfolgt ist.

Es muss dem Benutzer also zunächst eine Frage zusammen mit einer Anzahl von Antwortmöglichkeiten angezeigt werden. Der User wählt die Antwort aus, die für ihn am zutreffendsten ist. Darüber hinaus werden drei weitere UI-Elemente für die Bestätigung der Auswahl, die Anzeige der Ergebnisse und das Abschließen der Umfrageprozedur benötigt.

4.4.4.2. Das Aufgaben-Modell-Fragment

Zunächst wenden wir uns dem Aufgaben-Teilmodell des “Poll”-Patterns zu, aus dem das Aufgaben-Modell-Fragment (engl. Task Model Fragment“, TMF) des Musters konstruiert wird. Dieses wird letztendlich, wie alle anderen Modell-Fragmente auch, mittels der Beschreibungssprache PPSL spezifiziert und über das Beschreibungselement <Deployment> <PaMGIS> <ModelFragments> in die Pattern-Spezifikation eingebunden. Eine detaillierte Beschreibung von PPSL befindet sich in Kapitel 4.7.2.

Wie in Abbildung 18 gezeigt, ist das Wurzel-Element des zugrundeliegenden Aufgaben-Modells die abstrakte Aufgabe „poll“ (Umfrage) , welche aus den drei Unteraufgaben „vote“ (Abstimmung), „retrieve results“ (Ergebnisse anfordern) und „terminate“ (beenden) besteht.

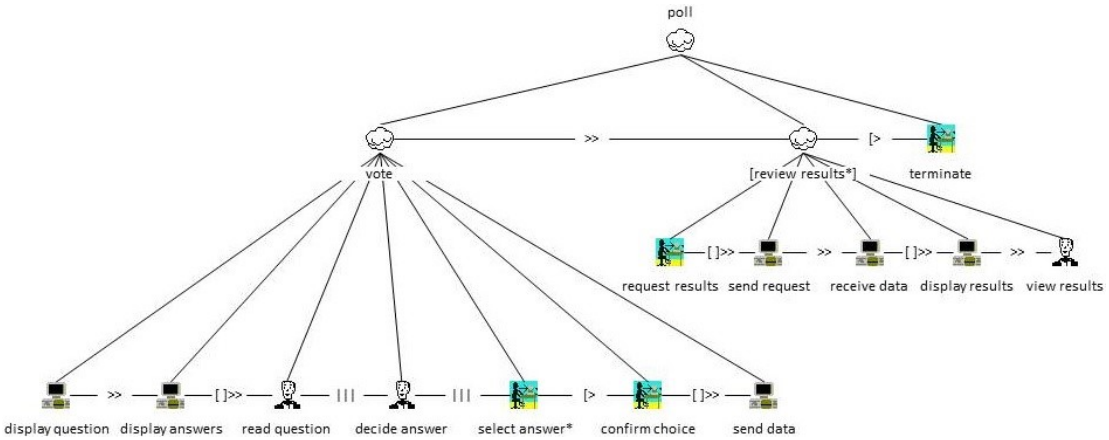


Abbildung 18 GESAMTES AUFGABEN-MODELL-FRAGMENT DES „POLL“-PATTERNS

Die Aufgabe „vote“ besteht ihrerseits aus untergeordneten Anwendungs-Tasks für die Anzeige der Frage und die zugehörenden Antwortmöglichkeiten, User-Tasks zum Lesen der Anzeige und zur Auswahl der gewünschten Antwort, Interaktions-Tasks zur Bestätigung der Antwort und zum Absenden der Auswahl sowie eine weitere Anwendungs-Aufgabe zur Übergabe der Daten an die der Anwendung zugrundeliegenden Business-Logik. Die Aufgabe

„retrieve results“ ist im Beispiel als optional definiert und muss deshalb nicht zwingend vom Benutzer ausgeführt werden. Sie umfasst eine Interaktions-Aufgabe zur Anforderung der Umfrageergebnisse, drei Anwendungs-Tasks zum Absenden dieses Auftrags an die Business-Logik, zum Empfangen der Ergebnisse und zu deren Anzeige sowie einer User-Aufgabe, um die Ergebnisse einzusehen. Die mit „terminate“ bezeichnete Interaktions-Aufgabe dient schließlich dazu, die Umfrageprozedur zu beenden.

In den folgenden Schritten wird dieses Task-Modell-Fragment soweit reduziert, dass nur noch die für die Generierung der Benutzeroberfläche benötigten Informationen enthalten sind. Die User-Tasks kennzeichnen unter anderem das kognitive Arbeitspensum der Benutzer, tragen aber ansonsten nicht wesentlich zum eigentlichen User-Interface bei. Deshalb können sie hier, wie in Abbildung 19 gezeigt, für die Zwecke von PaMGIS weggelassen werden.

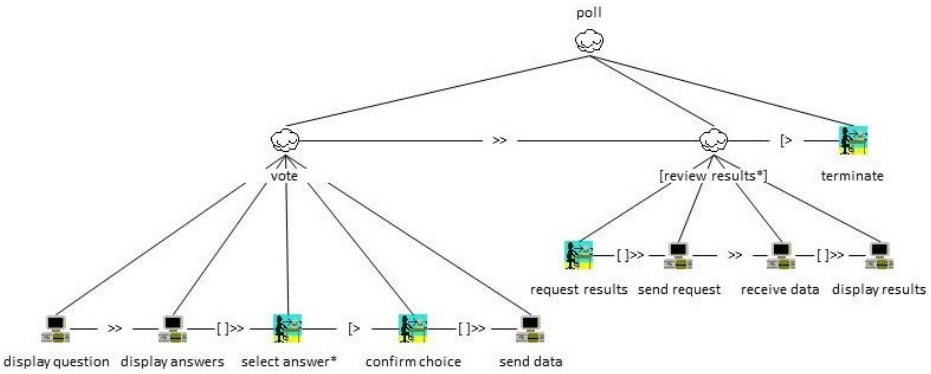


Abbildung 19 UM DIE USER-TASKS REDUZIERTES AUFGABEN-MODELL-FRAGMENT DES „POLL“-PATTERNS

Einige der verbliebenen Aufgaben verfügen über temporale Abhängigkeiten, die mittels des zeitlichen Operators *enabling with information passing* ([]>>) gekennzeichnet sind. Die durch diesen Operator verbundenen Aufgaben beziehen sich jeweils auf dieselben Daten und können durch jeweils einzelne UI-Elemente dargestellt werden. Aus diesem Grund werden die jeweiligen Anwendungs-Tasks aus dem TMF gestrichen. Im vorliegenden Beispiel sind dies die Aufgaben „display answers“, „send data“, „send request“ und „receive data“. Danach ergibt sich das in Abbildung 20 gezeigte Modell.

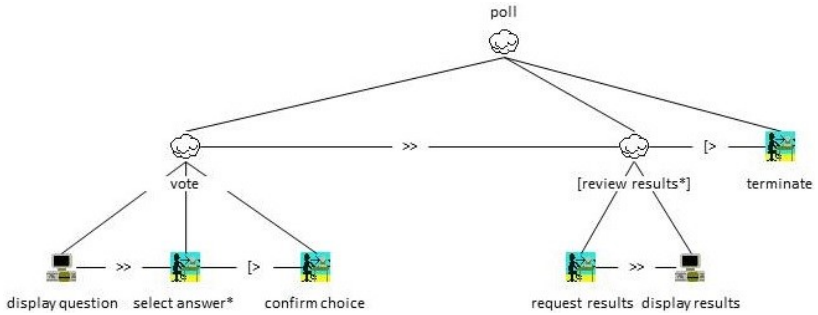


Abbildung 20 UM DIE REDUNDANTEN ANWENDUNG-TASKS REDUZIERTES AUFGABEN-MODELL-FRAGMENT DES „POLL“-PATTERNS

Um die Muster flexibler einsetzen zu können, kommt bei PaMGIS im Vergleich zu *ConcurTaskTrees* (CTT) [63] ein zusätzlicher Aufgabentyp zum Einsatz. Neben den bei CTT verwendeten abstrakten Aufgaben und Benutzer-, Anwendungs- und Interaktions-Aufgaben

unterstützt PaMGIS auch Dummy-Tasks. Dieser Aufgabentyp kommt ausschließlich in Aufgaben-Modell-Fragmenten vor, jedoch nie in den eigentlichen Aufgaben-Modellen. Die Dummy-Tasks dienen als Vorlagen für variable Teile eines TMF und werden ersetzt, sobald ein Pattern tatsächlich angewendet wird. Wenn ein PaMGIS-Benutzer ein Pattern, das Dummy-Tasks enthält, mit dem *Pattern Selection & Assignment*-Werkzeug auswählt und anwendet, wird automatisch ein Dialog geöffnet und nachgefragt, wie die Dummy-Tasks übernommen werden sollen. Dabei dient ihre vorgegebene Struktur als Konstruktionsplan für den sich ergebenden Aufgabentyp, die Position der Aufgabe im TMF und die jeweiligen temporalen Beziehungen zu anderen Aufgaben. Das auf diese Weise ergänzte TMF ist in Abbildung 21 dargestellt.

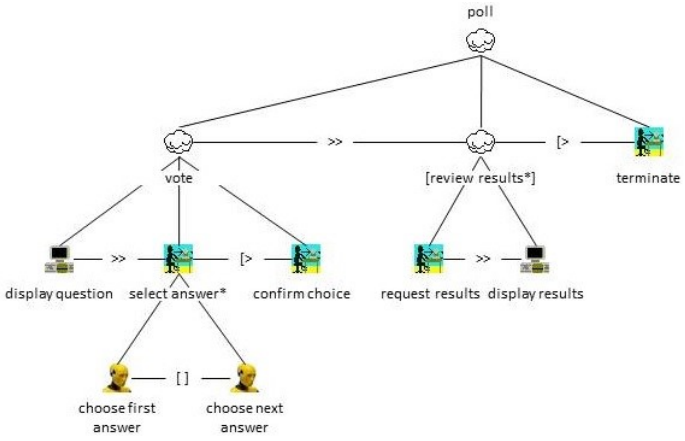


Abbildung 21 UM DUMMY-TASKS ERWEITERTES AUFGABEN-MODELL-FRAGMENT DES „POLL"-PATTERNS

Der PaMGIS-Benutzer kann den Namen der Dummy-Tasks interaktiv im vorher genannten Dialog nach Belieben ändern und entscheiden, wie oft die Anwendung des Dummy-Mechanismus wiederholt werden soll.

Das durch die Anwendung der Dummy-Tasks entstandene TMF wird in Abbildung 22 gezeigt. Im vorliegenden Beispiel wird die Dummy-Aufgabe „choose first answer" in „choose ‚excellent'"und „choose next answer" in „choose ‚fair'" umbenannt. Zudem wird der rechte Dummy ein zweites Mal angewendet und „choose ‚poor'" benannt.

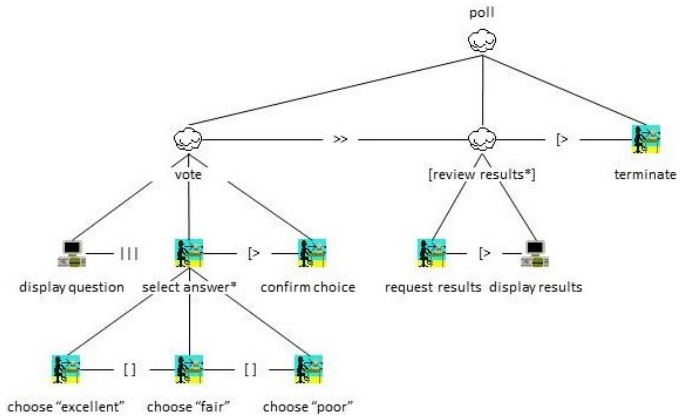


Abbildung 22 AUFGABEN-MODELL-FRAGMENT DES „POLL"-PATTERNS NACH AUFLÖSEN DER DUMMY-TASKS

Es ist zu beachten, dass das zum Muster gehörende Konzept-Modell-Fragment gleichermaßen Dummy-Konzepte enthalten kann, die in analoger Weise anzupassen sind.

Mit dem Muster abgespeichert wird natürlich die in Abbildung 21 veranschaulichte Version des TMF, das die Dummy-Tasks enthält. Dies ermöglicht dann dessen Anpassung im Rahmen der Anwendung des Patterns durch den PaMGIS-Benutzer. Das Task-Modell-Fragment wird hierzu mittels PPSL formalisiert beschrieben und im Beschreibungselement <Deployment> <PaMGIS> <ModelFragments> hinterlegt. Ein Ausschnitt der PPSL-Repräsentation des oben hergeleiteten TMF ist in Abbildung 23 dargestellt. Zur besseren Orientierung wurden dabei Zeilennummern eingefügt, die in der eigentlichen Spezifikation nicht enthalten sind. Die komplette Spezifikation des „Poll“-Patterns kann in Anhang I eingesehen werden.

```
01: <ModelFragment>
02:  <MDFR_Type>Task</MDFR_Type>
03:  <MDFR_FragmentID>"__TMF_0100_01"</MDFR_FragmentID>
04:  <MDFR_Label>"Poll"</MDFR_Label>
05:  <MDFR_Diagram>"siehe Abbildung 21 "</MDFR_Diagram>
06:  <MDFR_Fragment>
07:   <TMF_IncludesDummy>True</IncludesDummy>
08:   <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp
09:   <TMF_Content>
10:    <Subtask>
11:     <TaskID>"__TSK_0100_01_0001"</TaskID>
12:     <TaskName>"poll"</TaskName>
13:     <TaskDescription>""</TaskDescription>
14:     <TaskType>Abstraction</TaskType>
15:     <TaskOrigin>
16:      <TaskOriginPatternID>"PPT_0100_0001"</TaskOriginPatternID>
17:      <TaskOriginPatternName>"Poll"</TaskOriginPatternName>
18:      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
19:      <TaskOriginPatternRevision>"0000"</TaskOriginPatternRevision>
20:     </TaskOrigin>
21:     <Optional>False</Optional>
22:     <Iterative>False</Iterative>
23:     <Position>
24:      <Parent>
25:       <ParentID>""</ParentID>
26:       <ParentName>""</ParentName>
27:      </Parent>
28:      <SiblingLeft>
29:       <SiblingLeftID>""</SiblingLeftID>
30:       <SiblingLeftName>""</SiblingLeftName>
31:      </SiblingLeft>
32:      <SiblingRight>
33:       <SiblingRightID>""<SiblingRightID>
34:       <SiblingRightName>""</SiblingRightName>
35:      </SiblingRight>
36:     </Position>
37:     <TemporalOperator>Interleaving</TemporalOperator>
38:     <UIConcepts>
39:      <UIConcept>
40:       <CCPT_ConceptID>"__CPT_0100_01_0001"</CCPT_ConceptID>
41:       <CCPT_ConceptName>"Question"</CCPT_ConceptName>
42:      </UIConcept>
43:     </UIConcepts>
44:     <Subtasks>
45:      <Subtask>
46:       <TaskID>"__TSK_0100_01_0002"</TaskID>
47:       <TaskName>"vote"</TaskName>
……
```

Abbildung 23 AUSSCHNITT DER PPSL-REPRÄSENTATION DES TMF DES „POLL“-PATTERNS

Wenn das Muster ausgewählt wurde und angewendet wird, dann wird das TMF des Patterns in das Aufgaben-Modell, an dem gerade gearbeitet wird, an der zuvor bestimmten Stelle automatisch integriert. Dies geschieht, indem das Wurzel-Element des TMF mit seinem

neuen Eltern- und gegebenenfalls mit den vorhandenen Geschwister-Tasks verlinkt wird. Die Spezifikation des Wurzel-Elements beginnt in Abbildung 23 ab Zeile 10. In einem ersten Schritt wird der gesamte Inhalt des Beschreibungselements <TMF_Content> (siehe Zeile 09) in das Aufgaben-Modell kopiert. Anschließend werden ID und Name des neuen Eltern-Tasks in das <Parent>-Element des TMF-Wurzel-Elements eingetragen (siehe Zeilen 25 und 26). Falls vorhanden, wird dies auch in analoger Weise für die Geschwister-Tasks durchgeführt (siehe Zeilen 29 und 30 für die linke und Zeilen 33 und 34 für die rechte Geschwister-Aufgabe). Ferner müssen noch ID und Name des TMF-Wurzel-Elements in das <RightSibling>-Element der vorhergehenden und in das <LeftSibling>-Element der nachfolgenden Aufgabe kopiert werden. Die temporale Beziehung zwischen TMF-Wurzel-Element und seinem neuen rechten Geschwister wird nun noch mithilfe eines interaktiven Benutzerdialogs abgefragt und entsprechend in das <TemporalOperator>-Element (siehe Zeile 37) eingetragen. Jetzt muss nur noch die temporale Beziehung zum linken Geschwister spezifiziert werden. Ein Vorschlag hierfür ist im Beschreibungselement <TMF_ProposedTempOp> des TMF enthalten (siehe Zeile 08). Der PaMGIS-Benutzer kann entscheiden, ob dieser Vorschlag übernommen oder durch einen anderen Wert ersetzt werden soll. Zuletzt wird die gewählte temporale Beziehung im <TemporalOperator>-Element der Vorgängeraufgabe eingetragen.

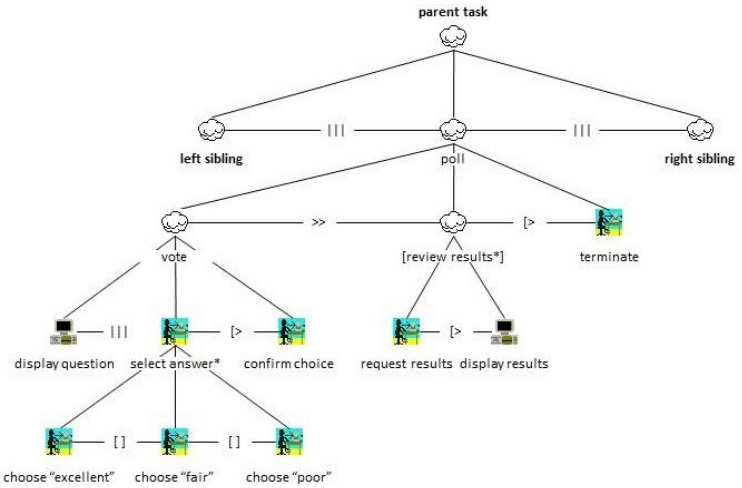


Abbildung 24 AUFGABEN-MODELL-FRAGMENT DES „POLL"-PATTERNS, DAS IN EIN BEISPIELHAFTES AUFGABEN-MODELL INTEGRIERT WURDE

In Abbildung 24 wird ein beispielhaftes Aufgaben-Modell, in welches das „Poll"-Pattern auf die oben beschriebene Art integriert wurde, skizziert. Die Aufgaben, die nicht aus dem TMF des „Poll"-Patterns stammen, sind hierbei durch fettgedruckte Task-Namen gekennzeichnet.

4.4.4.3. Das Konzept-Modell-Fragment

In diesem Unterkapitel wird gezeigt, wie das zum vorher beschriebenen TMF korrespondierende Konzept-Modell-Fragment (engl. „Concept Model Fragment", CMF) erstellt und mittels PPSL formal beschrieben wird.

Ein CMF beinhaltet die Spezifikationen aller Daten- und Interaktionselemente, die in der Benutzeroberfläche vorkommen. Deshalb müssen nun alle für das „Poll"-Patterns relevanten Konzepte identifiziert und die zugehörigen abstrakten Konzepttypen festgelegt werden.

Die im Beispiel verwendeten Typen sind in Tabelle 21 zusammengefasst und kurz erklärt. Eine vollständige und ausführliche Beschreibung der in PaMGIS zum Einsatz kommenden Konzepte ist in Kapitel 4.6.2 zu finden.

Tabelle 21 LISTE DER IM „POLL"-PATTERN VERWENDETEN KONZEPTE

Element	Beschreibung
<Activator>	Dient dazu, ein UI-Element zu aktivieren oder einen Funktionsaufruf zu initiieren.
<DataOutput>	Dient dazu, Informationen für den Benutzer wahrnehmbar zu machen, z. B. durch Anzeige auf dem Bildschirm oder die Wiedergabe eines Audio-Streams.
<SingleChoice>	Dient dazu, genau eine aus einer Menge von Optionen auszuwählen.
<ChoiceItem>	Beschreibt eine der Optionen, die mithilfe eines SingleChoice-Konzepts ausgewählt werden können.
<Navigator>	Dient dazu, zu einem anderen Fenster, Bildschirm oder Dialog zu gelangen.

Mit Blick auf das TMF wird klar, dass für die Realisierung des „Poll"-Patterns insgesamt acht Konzepte benötigt werden. Wie in Abbildung 25 gezeigt wird, handelt es sich dabei um:

1. ein <DataOutput>-Element, um die Frage anzuzeigen,
2. ein <SingleChoice>-Element, um die Antwort auszuwählen,
3. ein <ChoiceItem>-Element für die erste Antwortmöglichkeit (erster Dummy),
4. ein weiteres <ChoiceItem> für die zweite Antwortmöglichkeit (zweiter Dummy),
5. ein <Activator>-Element, um die Auswahl zu bestätigen,
6. ein weiteres <Activator>-Element, um das Umfrageergebnis anzufordern,
7. ein <DataOutput>-Element zur Anzeige des Umfrageergebnisses und
8. ein <Navigator>-Element, um die Umfrageprozedur zu beenden.

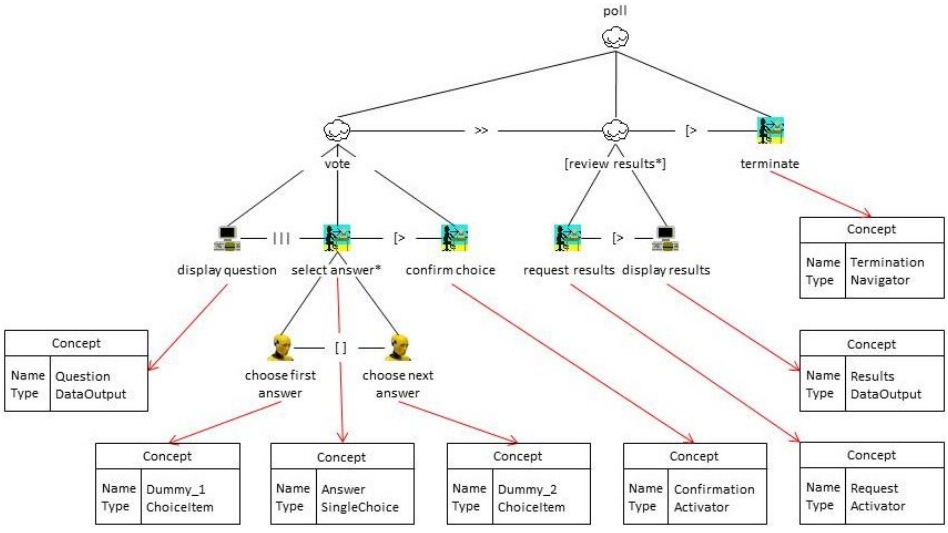


Abbildung 25 ABLEITUNG DER BENÖTIGTEN KONZEPTE AUS DEM TMF DES „POLL"-PATTERNS

Die PPSL-Beschreibungen dieser Konzepte werden im Konzept-Modell-Fragment des Musters aufgelistet. Ein Ausschnitt des CMF wird in Abbildung 26 veranschaulicht. Auch hier wurden wieder zur besseren Lesbarkeit Zeilennummern eingefügt, die im eigentlichen CMF nicht enthalten sind.

```
01: <ModelFragment>
02:  <MDFR_Type>Concept</MDFR_Type>
03:  <MDFR_FragmentID>"__CMF_0100_01"</MDFR_FragmentID>
04:  <MDFR_Label>"Concepts for poll pattern"</MDFR_Label>
05:  <MDFR_Fragment>
06:   <CMF_Content>
07:    <Concept>
08:     <CCPT_ConceptID>"__CPT_0100_01_0001"</CCPT_ConceptID>
09:     <CCPT_ConceptName>"Question"</CCPT_ConceptName>
10:     <CCPT_Description>"Question to be answered by the user"</CCPT_Description>
11:     <CCPT_Label>""</CCPT_Label>
12:     <CCPT_Perceptible>True</CCPT_Perceptible>
13:     <CCPT_Enabled>True</CCPT_Enabled>
14:     <CCPT_Required>True</CCPT_Required>
15:     <CCPT_ConceptType>DataOutput</CCPT_ConceptType>
16:     <CCPT_DataType>"String"</CCPT_DataType>
17:     <CCPT_ConceptOrigin>
18:      <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
19:      <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
20:      <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
21:      <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
22:     </CCPT_ConceptOrigin>
23:     <CCPT_Preconditions></CCPT_Preconditions>
24:     <CCPT_Postconditions><CCPT_Postconditions>
25:     <CCPT_DataLink></CCPT_DataLink>
26:     <CCPT_TaskLinks>
27:      <CCPT_TaskLink>
28:       <CCPT_TaskID>"__TSK_0100_01_0005"</CCPT_TaskID>
29:       <CCPT_TaskName>"display question"</CCPT_TaskName>
30:      </CCPT_TaskLink>
31:     </CCPT_TaskLinks>
32:    </Concept>
33:    <Concept>
34:     <CCPT_ConceptID>"__CPT_0100_01_0002"</CCPT_ConceptID>
35:     <CCPT_ConceptName>"Answer"</CCPT_ConceptName>
.....
```

Abbildung 26 AUSSCHNITT DER PPSL-REPRÄSENTATION DES CMF DES „POLL“-PATTERNS

Die Spezifikation des ersten Konzepts, das zur Anzeige der Frage dient, erfolgt in den Zeilen 07 bis 32, die des zweiten beginnt ab Zeile 33. Für jedes Konzept sind Verknüpfungen zu denjenigen Aufgaben im TMF, die dieses Konzept benötigen, hinterlegt (siehe Zeilen 26 bis 31). In umgekehrter Weise verfügen die betreffenden Task-Spezifikationen über Verlinkungen zu den zugehörenden Konzepten (siehe Zeilen 38 bis 43 in Abbildung 23 auf Seite 130).

Bei der Anwendung des Musters durch das PaMGIS-Framework wird der Benutzer in einem Dialog gefragt, ob die im CMF vorhandenen Konzepte in das Konzept-Modell übernommen werden sollen. Wird dies bejaht, so werden die Konzepte, d.h. der Inhalt des Beschreibungselements <CMF_Content>, einfach in das Konzept-Modell kopiert.

4.4.4.4. Die Dialog-Modell-Fragmente

Dialog-Modell-Fragmente (engl. „Dialog Model Fragment“, DMF) werden auf der Grundlage des TMF spezifiziert. Unter Berücksichtigung des für die Benutzeroberfläche vorgesehenen Benutzungskontexts, also beispielsweise der Möglichkeiten, die ein bestimmtes Endgerät bietet, wird dieses in Teilmengen von Aufgaben unterteilt. Diese stellen dann die Dialoge dar. Hierfür liefert die hierarchische Struktur des Aufgaben-Modells, d. h. die verschiedenen Unterbäume und die temporalen Beziehungen zwischen den einzelnen Aufgaben wertvollen Input. Zusätzlich müssen noch die Übergänge zwischen den Dialogen definiert werden.

Es ist anzumerken, dass Muster über mehrere, für die verschiedensten Benutzungskontexte angepasste Dialog-Modell-Fragmente besitzen können.

Die im PaMGIS-Framework zum Einsatz kommenden Dialog-Modelle und Dialog-Modell-Fragmente werden mittels Dialoggraphen modelliert, wie sie auch bei der an der Universität Rostock entworfenen modellbasierten Entwicklungsumgebung TADEUS (siehe Kapitel 2.5.1.12) verwendet werden.

In Abbildung 27 wird ein Dialog-Graph für den Einsatz des „Poll“-Patterns auf einem Desktop-Computer mit großem Bildschirm gezeigt. Es handelt sich um einen einzelnen Dialog, der alle Aufgaben des TMF (siehe Abbildung 21 auf Seite 129) in sich vereint.

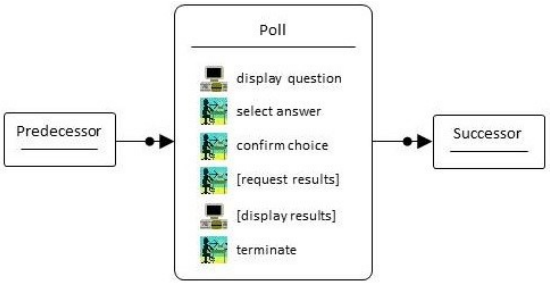


Abbildung 27 DIALOG-MODELL-FRAGMENT DES „POLL“-PATTERNS FÜR DESKTOP-COMPUTER MIT GROßEM BILDSCHIRM

Die PPSL-Repräsentation dieses DMF wird in Abbildung 28 dargestellt. Die eigentliche Spezifikation des Dialogs beginnt in Zeile 10. Das <Position>-Element (siehe Zeilen 16 bis 39) kann Informationen über potenzielle Vorgänger- und Nachfolger-Dialoge aufnehmen. Die entsprechenden Beschreibungselemente werden gefüllt, wenn das Muster angewendet wird. Insbesondere müssen, falls dies nötig beziehungsweise gewünscht ist, die Trigger für die Übergänge zwischen den Dialogen definiert werden. Hierfür dienen die Elemente <SUCC_Trigger> (Zeilen 33 bis 36) für die Hin- und gegebenenfalls <PRED_Trigger> (Zeilen 22 bis 25) für die Rückrichtung. Dieser Prozess kann nicht vollständig automatisiert ablaufen und muss durch den PaMGIS-Benutzer interaktiv gesteuert werden. Diejenigen Aufgaben aus dem TMF, die im vorliegenden Dialog enthalten sein sollen, werden mithilfe des Beschreibungselements <DLG_Tasks> (Zeilen 40 bis 56) spezifiziert. Dabei muss nicht zwingend jede einzelne Aufgabe angegeben werden. Mit dem <DLG_Processing>-Element (siehe Zeilen 44, 49 und 54) kann mit dem Wert *Exclusive* angezeigt werden, dass nur die betreffende Aufgabe selbst zum Dialog gehören soll, während der Wert *Recursive* dafür sorgt, dass neben der eigentlichen Aufgabe auch alle zugehörenden Unteraufgaben mit aufgenommen werden sollen.

```
01: <ModelFragment>
02:  <MDFR_Type>Dialog</MDFR_Type>
03:  <MDFR_FragmentID>"__DMF_0100_01"</MDFR_FragmentID>
04:  <MDFR_Label>"Poll"</MDFR_Label>
05:  <MDFR_Purpose>"Dialog Model Fragment for large screens"</MDFR_Purpose>
06:  <MDFR_Diagram>"siehe Abbildung 27 "</MDFR_Diagram>
07:  <MDFR_Fragment>
08:   <DMF_ContextModelReferences></DMF_ContextModelReferences>
09:   <DMF_Content>
10:    <Dialog>
11:     <DialogID>"__DLG_0100_01_01"</DialogID>
12:     <DialogName>"Poll"</DialogName>
13:     <DialogDescription>"Poll for large screens"</DialogDescription>
14:     <DialogLabel>"Poll"</DialogLabel>
15:     <DialogType>Modal</DialogType>
16:     <Position>
17:      <Predecessors>
18:       <Predecessor>
19:        <PRED_DialogID>""</PRED_DialogID>
20:        <PRED_DialogName>""</PRED_DialogName>
21:        <PRED_TransitionType>Sequential</SUCC_TransitionType>
22:        <PRED_Trigger>
23:         <PRED_ConceptID>""</PRED_ConceptID>
24:         <PRED_ConceptName>""</PRED_ConceptName>
25:        </PRED_Trigger>
26:       </Predecessor>
27:      <Predecessors>
28:      <Successors>
29:       <Successors>
30:        <SUCC_DialogID>""</SUCC_DialogID>
31:        <SUCC_DialogName>""</SUCC_DialogName>
32:        <SUCC_TransitionType>Sequential</SUCC_TransitionType>
33:        <SUCC_Trigger>
34:         <SUCC_ConceptID>"__CPT_0100_01_0006"</SUCC_ConceptID>
35:         <SUCC_ConceptName>"Termination"<SUCC_ConceptName>
36:        </SUCC_Trigger>
37:       </Successor>
38:      </Successors>
39:     </Position>
40:     <DLG_Tasks>
41:         <DLG_Task>
42:          <DLG_TaskID>"__TSK_0100_01_0002"</DLG_TaskID>
43:          <DLG_TaskName>"vote"</DLG_TaskName>
44:          <DLG_Processing>Recursive</DLG_Processing>
45:         <DLG_Task>
46:         <DLG_Task>
47:          <DLG_TaskID>"__TSK_0100_01_0003"</DLG_TaskID>
48:          <DLG_TaskName>"review results"</DLG_TaskName>
49:          <DLG_Processing>Recursive</DLG_Processing>
50:      <DLG_Task>
51:      <DLG_Task>
52:       <DLG_TaskID>"__TSK_0100_01_0004"</DLG_TaskID>
53:       <DLG_TaskName>"terminate"</DLG_TaskName>
54:       <DLG_Processing>Exclusive</DLG_Processing>
55:      </DLF_Task>
56:     </DLG_Tasks>
57:    </Dialog>
58:   </DMF_Content>
59:  </MDFR_Fragment>
60: </ModelFragment>
```

Abbildung 28 AUSSCHNITT DER PPSL-REPRÄSENTATION DES DMF DES „POLL“-PATTERNS

Ein weiteres Beispiel für ein DMF für das „Poll“-Pattern ergibt sich bei der Verwendung von mobilen Endgeräten mit vergleichsweise geringer Bildschirmgröße, also etwa für Smartphones. Durch die Größenbeschränkung der Anzeige ist es hierbei nicht möglich oder nicht zweckmäßig alle UI-Elemente gleichzeitig darzustellen. Dies kann durch die Verwendung von einer Sequenz von zwei Dialogen gelöst werden, so wie dies in Abbildung 29 gezeigt wird.

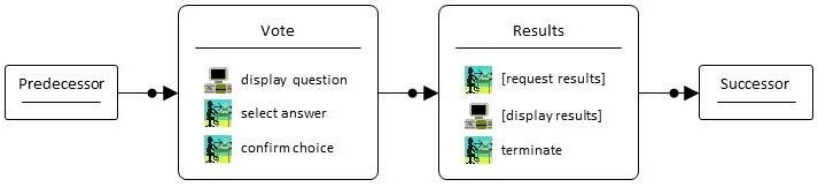


Abbildung 29 DIALOG-MODELL-FRAGMENT DES „POLL"-PATTERNS FÜR MOBILE ENDGERÄTE MIT KLEINEM BILDSCHIRM

Die Frage, welche Aufgaben in welchen Dialog aufgenommen werden sollen, lässt sich am besten mit einem Blick auf die Struktur des TMF in Abbildung 21 beantworten. In den meisten Fällen ist es erfolgversprechend diejenigen Aufgaben, die sich in ein und demselben Unterbaum befinden, in einem Dialog zu bündeln. Im vorliegenden Fall wurden die Tasks des Teilbaumes, der mit der Aufgabe „vote" beginnt, in einem ersten Dialog namens „Vote" zusammengefasst. Ein zweiter Dialog mit Namen „Results" nimmt die Aufgaben des Teilbaums ab Task „review results" auf. Eigentlich würde auf diese Weise noch ein dritter Dialog entstehen, der aber lediglich die noch verbliebene Aufgabe „terminate" beinhalten würde. Da dies keinen Sinn macht, wurde diese Aufgabe mit in den „Results"-Dialog aufgenommen. Der Übergang vom „Vote"- zum „Results"-Dialog erfolgt, sobald die Aufgabe „confirm choice" abgearbeitet wurde. Die Einbindung des DMF erfolgt bei der Anwendung des Musters durch Kopieren des gesamten Inhalts des Beschreibungselements <DMF_Content> in das Dialog-Modell, das gerade erstellt wird. Danach müssen noch die <Position>-Elemente des ersten und des letzten Dialogs aus dem DMF durch den PaMGIS-Benutzer an die Gegebenheiten des Dialog-Modells angepasst werden. Die PPSL-Repräsentation des DMF kann der Spezifikation des „Poll"-Patterns in Anhang I entnommen werden.

4.4.4.5. Die Ableitung der Benutzerschnittstellen

Bei der Generierung der Benutzerschnittstelle muss zunächst die Struktur des abstrakten User-Interface (AUI) aufgebaut werden. Dies geschieht auf Basis des Dialog-Modells. Die Aufgaben, die in den Dialogen zusammengefasst sind, besitzen, wie zuvor beschrieben, Verweise auf die zugehörenden Konzepte.

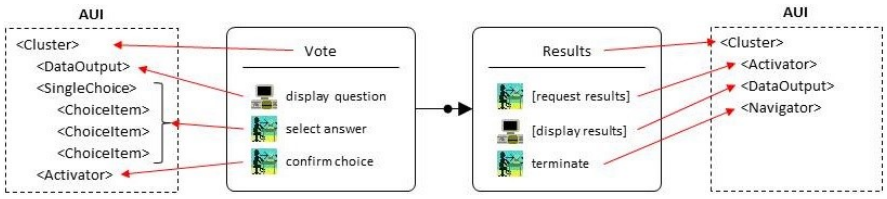


Abbildung 30 ABLEITUNG DES AUI AUS DEM DIALOG-MODELL FÜR MOBILE ENDGERÄTE MIT KLEINEM BILDSCHIRM

Wie in Abbildung 30 ersichtlich, wird für jeden Dialog ein <Cluster>-Element erzeugt, in das die Konzepte der ihm zugeordneten Aufgaben kopiert werden. Das erste <Cluster> für den „Vote"-Dialog besteht also aus einem <DataOutput>-Element für die Anzeige der zu beantwortenden Frage, einem <SingleChoice>-Element, dem im Beispiel drei <ChoiceItem>-Elemente mit den vorgegebenen Antwortmöglichkeiten zugeordnet sind und einem <Activator>-Element, mit dem die getroffene Auswahl bestätigt und der Übergang zu nachfolgenden Dialog initiiert wird. Das zweite <Cluster> steht für den „Results"-Dialog und verfügt über ein <Activator>-Element zur Aktualisierung der Ergebniswerte, ein

<DataOutput>-Element für die Anzeige der Ergebnis-Grafik und einem <Navigator>-Element zum Beenden der Umfrageprozedur. Ein Auszug aus der XML-Repräsentation dieses AUI wird in Abbildung 31 gezeigt. Wie schon bei den Modell-Fragmenten wurden auch hier Zeilennummern zur besseren Orientierung eingefügt.

```
01: <Cluster>
02:  <ClusterID=>"__CLS_01_0001"</ClusterID>
03:  <ClusterName>"Vote"</ClusterName>
04:  <DataOutput>
05:   <ConceptID>"__CPT_0100_01_0001"</ConceptID>
06:   <ConceptName>"Question"</ConceptName>
07:  </DataOutput>
08:  <SingleChoice>
09:   <ConceptID>"__CPT_0100_01_0002"</ConceptID>
10:   <ConceptName>"Answer"</ConceptName>
11:   <ChoiceItems>
12:    <ChoiceItem>
13:     <ConceptID>"__CPT_0100_01_0007"</ConceptID>
14:     <ConceptName>"Answer1"</ConceptName>
15:    </ChoiceItem>
16:    <ChoiceItem>
17:     <ConceptID>"__CPT_0100_01_0008"</ConceptID>
18:     <ConceptName>"Answer2"</ConceptName>
19:    </ChoiceItem>
20:    <ChoiceItem>
21:     <ConceptID>"__CPT_0100_01_0009"</ConceptID>
22:     <ConceptName>"Answer3"</ConceptName>
23:    </ChoiceItem>
24:   </ChoiceItems>
25:  </SingleChoice>
26:  </Activator>
27:   <ConceptID>"__CPT_0001_01_0003"</ConceptID>
28:   <ConceptName>"Confirmation"</ConceptName>
29:  </Activator>
30: </Cluster>
31: <Cluster>
32:  <ClusterID>"__CLS_01_0002"</ClusterID>
33:  <ClusterName>"Results"</ClusterName>
34:  <Activator>
35:   <ConceptID>"__CPT_0001_01_0004"</ConceptID>
36:   <ConceptName>"Request"</ConceptName>
37:  </Activator>
38:  <DataOutput>
39:   <ConceptID>"__CPT_0001_01_0005"</ConceptID>
40:   <ConceptName>"Results"</ConceptName>
41:  </DataOutput>
42:  <Navigator>
43:   <ConceptID>"__CPT_0001_01_0006"</ConceptID>
44:   <ConceptName>"Termination"</ConceptName>
45:  </Navigator>
46: </Cluster>
```

Abbildung 31 AUSZUG AUS DEM AUI DES „POLL“-PATTERNS FÜR MOBILE ENDGERÄTE MIT KLEINEM BILDSCHIRM

Im nächsten Schritt wird das AUI in ein konkretes User-Interface (CUI) überführt. Dabei werden die AUI-Elemente unter Berücksichtigung des ausgewählten Kontext-Modells durch die zu verwendenden CUI-Elemente ersetzt. Insbesondere gibt hierbei das Toolkit-Modell darüber Aufschluss, welche Typen von CUI-Elementen von der Zielsprache überhaupt unterstützt werden. Die Entscheidung, welche AUI- durch welche CUI-Elementtypen ersetzt werden, wird entweder durch entsprechende Entscheidungstabellen oder durch Auswahl in einem interaktiven Dialog durch den PaMGIS-User gefällt. Im folgenden Beispiel soll die Realisierung der Benutzerschnittstelle mittels *Hyper Text Markup Language* (HTML) erfolgen. Die Ersetzung wird in Abbildung 32 skizziert.

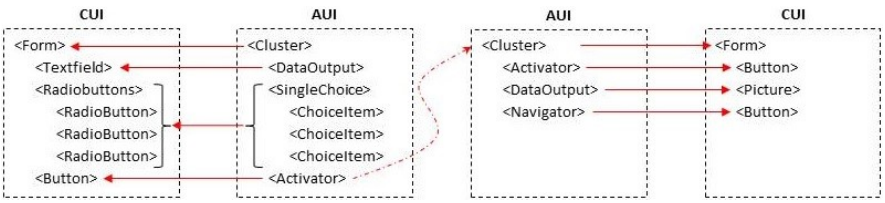


Abbildung 32 ABLEITUNG DES CUI AUS DEM AUI FÜR MOBILE ENDGERÄTE MIT KLEINEM BILDSCHIRM ZUR REALISIERUNG IN HTML

Dabei werden die Dialoge, d. h. die <Cluster> durch Formular-CUI-Elemente ersetzt, das <DataOutput>-Element für die zu beantwortende Frage durch ein Textfeld und das Auswahl-Element <SingleChoice> mit den zugehörenden Antwortmöglichkeiten durch Radiobuttons. Die <Navigator>- und <Activator>-Elemente werden gleichermaßen in Buttons umgewandelt und das verbliebene <DataOutput>-Element in eine Grafik.

Letztlich muss noch das CUI in das finale User-Interface (FUI) in der Zielsprache HTML überführt werden. Dabei werden die zwei <Form>-Elemente in <form target="_blank">-Konstrukte, die beide die jeweiligen UI-Elemente der „Vote"- beziehungsweise „Results"-Dialoge beinhalten, transformiert. Das Textfeld wird durch das HTML-Tag <p>, die Radiobuttons durch <input type="radio">- und die Buttons mithilfe von <button type="submit">-Tags realisiert. Die Grafik wird mittels <img>-Tag dargestellt. Abbildung 33 illustriert die fertigen HTML-Dialoge.

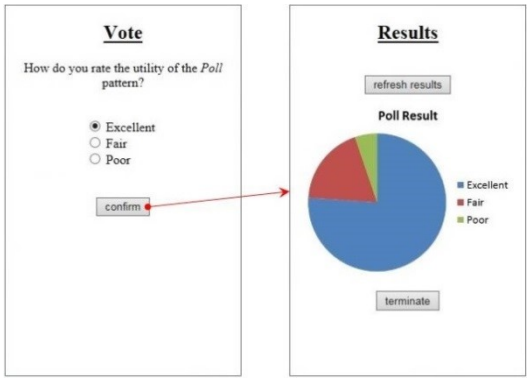


Abbildung 33 IN HTML REALISIERTES FUI FÜR MOBILE ENDGERÄTE MIT KLEINEM BILDSCHIRM

4.5. Prototypische Implementierung

Wesentliche Teilaspekte des PaMGIS-Frameworks wurden im Rahmen von zwei von mir betreuten Masterarbeiten an der Hochschule Augsburg prototypisch entwickelt.

Zum einen wurde auf Basis des zu diesem Zeitpunkt aktuellen Spezifikationsstands das Pattern-Repository und ein Werkzeug zur Erstellung und Verwaltung von PPSL-konformen Patterns und Pattern-Sprachen implementiert [201]. Mit einem maskenorientierten Pattern-Editor können Muster erfasst und geändert sowie Beziehungen zwischen den Patterns definiert werden. Jedem Benutzer steht ein privater Workspace für die persistente Speicherung seiner Arbeitsergebnisse zur Verfügung. Neue Muster können als Vorschläge eingereicht, vom Administrator freigegeben und allen anderen Benutzern zugänglich gemacht werden.

Die Realisierung erfolgte unter Verwendung von JavaFX[186] und Hibernate[187] im Zusammenspiel mit MySQL[188].

Zwei Screenshots dieser als *PPSL-Pattern-Management-Tool* (PPMT) bezeichneten Anwendung sind in Abbildung 34 und Abbildung 35 zu sehen.

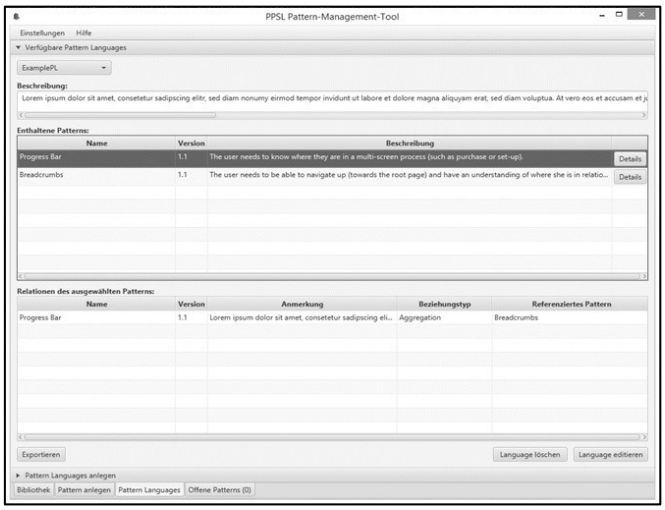


Abbildung 34 SCREENSHOT DES BILDSCHIRMS ZUR VERWALTUNG VON PATTERNS UND PATTERN-SPRACHEN

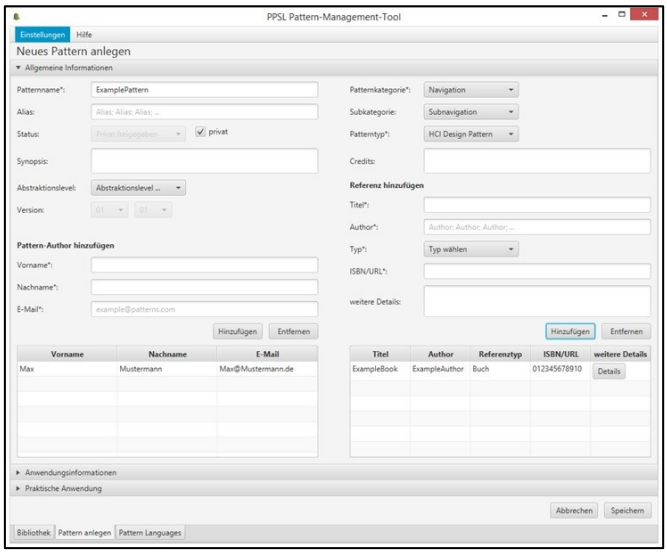


Abbildung 35 SCREENSHOT DES BILDSCHIRMS ZUR NEUANLAGE VON PATTERNS

Zum anderen handelt es sich um den *PaMGIS Pattern Code Generator* (PaCo) für die Generierung interaktiver Benutzeroberflächen aus PaMGIS-Modellen für die Android[189]-

[186] JavaFX ist ein Framework von Oracle Corp. zur Erstellung plattformübergreifender Java-Applikationen
[187] Hibernate ist ein objektrelationales Persistenz-Framework der zur Firma Red Hat gehörenden JBOSS Inc.
[188] MySQL ist ein relationales Datenbanksystem von Oracle Corp.

Plattform. Diese Komponente erlaubt die automatisierte Generierung und Transformation von abstrakten (AUI), konkreten (CUI) und finalen (FUI) Benutzeroberflächenmodellen [202].

Implementiert wurde PaCo auf Basis von JavaFX und JAXB[190].

In Abbildung 36 wird ein Screenshot der Benutzeroberfläche des PaCo-Generators und in Abbildung 37 ein Bildschirm einer mit PaCo generierten Android-App für Fahrplanauskünfte gezeigt.

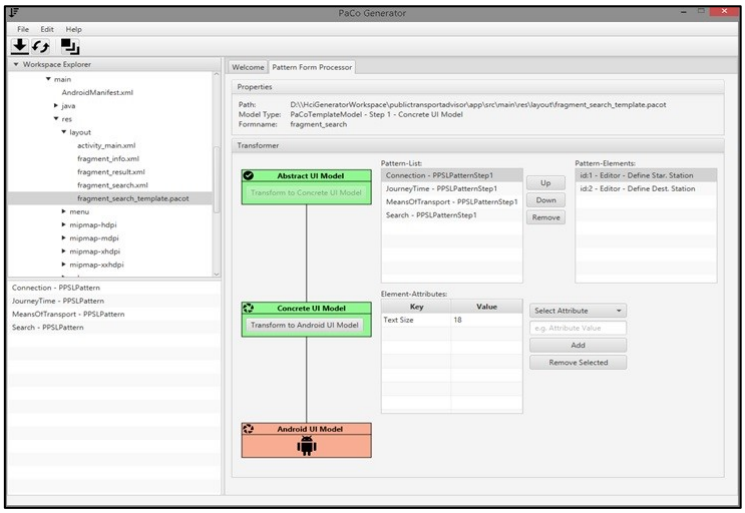


Abbildung 36 SCREENSHOT DES PACO-GENERATORS ZUR AUTOMATISIERTEN ERZEUGUNG VON ANDROID-APPS

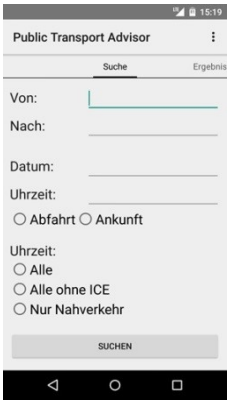


Abbildung 37 SCREENSHOT EINER MIT DEM PACO-GENERATOR ERZEUGTEN ANDROID-APP ZUR FAHRPLANAUSKUNFT

Weitere Details zu den beiden Werkzeugen sind in [203] zusammengefasst.

---

189 Android ist ein Betriebssystem und eine Software-Plattform für mobile Endgeräte der von der Google LLC gegründeten Open Handset Alliance

190 Java Architecture for XML Binding (JAXB) ist eine Programmierschnittstelle für Java zum Zugriff auf XML-konforme Daten

4.6. Detailaspekte des modellgetriebenen Anteils

In diesem Kapitel wird zunächst eine Übersicht über die Werkzeuge des modellgetriebenen Teils von PaMGIS gegeben. Anschließend wird der Aufbau der verschiedenen Modelle im Detail beschrieben.

4.6.1. Übersicht über die Modell-Werkzeuge

Ein Überblick über die gesamte Palette der durch die PaMGIS-Werkzeuge zur Verfügung gestellten Funktionalitäten und deren Zusammenhänge wird in Abbildung 38 illustriert.

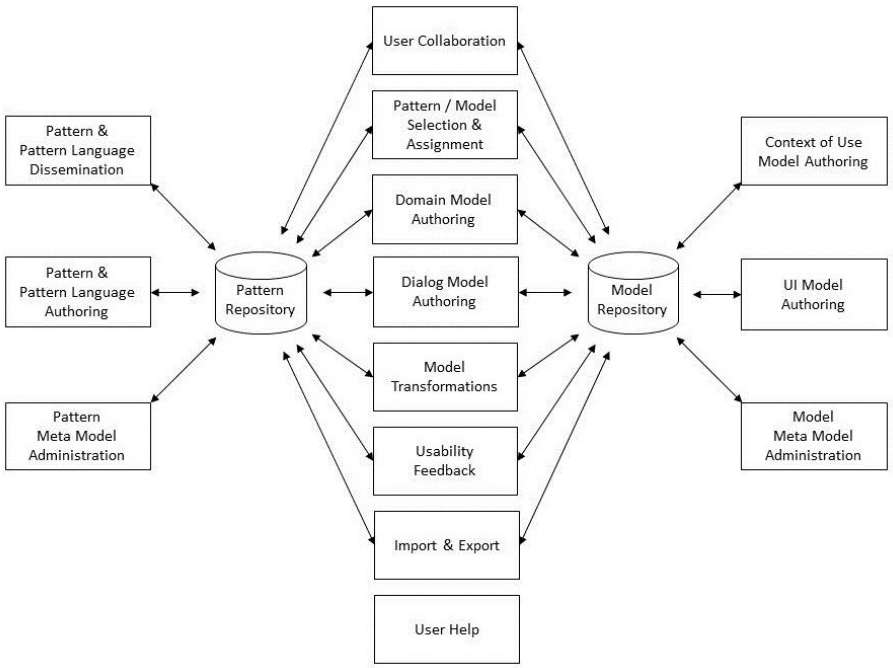


Abbildung 38 LOGISCHE ÜBERSICHT ÜBER DIE FUNKTIONEN DER PAMGIS-WERKZEUGE

Hierbei handelt es sich um eine logische Betrachtungsweise, d.h. dass nicht jede der gezeigten Funktionalitäten zwingend in einem separaten Werkzeug implementiert ist, sondern auch durchaus mehrere in einem gemeinsamen Tool zusammengefasst sein können.

Während auf der linken Seite von Abbildung 38 die Funktionen für das Pattern-Repository dargestellt sind, werden auf der rechten Seite die Funktionalitäten hinsichtlich des Modell-Repository gezeigt. In der Mitte sind alle Funktionen, die sowohl für Muster als auch für die Modelle verfügbar sind, gezeigt. Im vorliegenden Kapitel wird auf die Werkzeuge des modellgetriebenen Teils von PaMGIS näher eingegangen. Die Pattern-Tools werden in Kapitel 4.7.1 erläutert. Zur Vereinfachung werden hierbei jeweils die englischen Bezeichnungen verwendet.

Die Funktion *Model Meta Modell Administration* wird von der Rolle des Framework-Administrators genutzt. Er kann damit die bestehenden Metamodelle, die einer Versionierung unterliegen, editieren. Soll ein Metamodell verändert werden, so wird stets zuerst eine Kopie des betreffenden Modells erzeugt, die dann editiert und anschließend als neue Ver-

sion gespeichert wird. Mit der Freigabe steht diese dann für die Rollen des Modell-Administrators und Power-Users zur Verfügung. Aus Konsistenzgründen werden keine Versionen von Metamodellen gelöscht, aber sie können zurückgezogen werden. In diesem Fall stehen sie für eine neuerliche Verwendung nicht mehr bereit.

Hinter den Funktionen *Domain Model Authoring*, *Context of Use Authoring*, *Dialog Model Authoring* und *UI Model Authoring* verbergen sich Editoren die für die jeweils im Namen genannten Modelltypen. Sie können, wie in Kapitel 4.4.2 beschrieben, durch den Modell-Administrator bzw. Power-User dazu genutzt werden, um neue archetypische bzw. benutzerdefinierte Modelle anzulegen und bestehende Modelle zu verändern. Wie die Metamodelle unterliegen die archetypischen Modelle einer Versionierung. Auch sie werden nicht gelöscht, können aber durch den Modell-Administrator zurückgezogen werden. Hingegen können benutzerdefinierte Modelle sowohl gelöscht, als auch ohne Anlegen einer neuen Version beliebig verändert werden. Darüber hinaus unterstützen die Authoring-Tools Browsing und Suche im Modell-Repository. Hierbei bietet Browsing die Möglichkeit zur Auflistung der vorhandenen Modelle und der Anzeige von Detailinformationen zu den Listeneinträgen, die ausgewählt wurden. Die Suche-Funktionalität umfasst das Suchen nach Schlüsselworten, Volltext-Suche und eine erweiterte Suche, bei der verschiedene Suchfelder definiert und logisch verknüpft werden können.

Pattern / Model Selection & Assignment bietet zweierlei Funktionalitäten. Einerseits erlaubt es, Muster aus dem Pattern-Repository herauszusuchen und bei der Erstellung der Modelle zu verwenden. Bei der Anwendung eines Musters werden die in der Pattern-Spezifikation enthaltenen Modell-Fragmente wie Bausteine in die gerade bearbeiteten Modelle eingebaut. Andererseits können hiermit auch archetypische oder benutzerdefinierte Modelle lokalisiert, ausgewählt und zum aktuellen PaMGIS-Projekt hinzugefügt werden.

Die *Model Transformations* sind Hilfsmittel zur Generierung des AUI-Modells und dessen schrittweisen Transformation in das CUI- und schließlich FUI-Modell (siehe Kapitel 4.6.3). Hierbei besteht die Möglichkeit, Informationen aus zuvor ausgewählten Patterns einfließen zu lassen.

Die Funktion *Usability Feedback* dient dazu, Erkenntnisse zur Gebrauchstauglichkeit der Benutzeroberfläche, die beispielsweise durch Usability-Evaluation ermittelt wurden, bei den betreffenden Patterns- und Modell-Spezifikationen zu hinterlegen. So kann der Bedarf für entsprechende Überarbeitung erkannt und diese unter Berücksichtigung des Feedbacks durchgeführt werden.

Die Funktionalität der *User Collaboration* unterstützt den Informationsaustausch zwischen den verschiedenen PaMGIS-Benutzern durch Möglichkeiten zur Diskussion und Kommentierung. Dabei wird zwischen einer geschlossenen und einer offenen Zusammenarbeit unterschieden. Bei der geschlossenen Kollaboration findet ein Austausch von Nachrichten zwischen zwei oder mehreren Benutzern statt. Dabei haben nur die jeweils Beteiligten Zugriff auf die entsprechenden Inhalte. Die offene Kommunikation findet mittels eines Diskussions-Forums statt. Hierbei können die Benutzer beliebige Beiträge posten. Diese können von allen anderen Usern eingesehen und kommentiert werden. Ein weiteres Hilfsmittel zur Kollaboration ist die Möglichkeit für Power-User, ein von ihnen erstelltes Modell zur Aufnahme in die Menge der archetypischen Modelle vorzuschlagen.

Die *Import & Export* Funktionalität steht nur für die Administratoren-Rollen zur Verfügung. Damit werden sie in die Lage versetzt, einzelne oder mehrere Modell-Spezifikationen,

gegebenenfalls mit Durchführung von erforderlichen Transformationen in ein bestimmtes Zielformat, in das PaMGIS-Framework zu übernehmen oder für eine Verwendung ausserhalb von PaMGIS zur Verfügung zu stellen.

Nicht zuletzt versorgt *User Help* die Benutzer bezüglich aller PaMGIS-Komponenten und Funktionalitäten mit entsprechenden Hilfe-Informationen.

Die Beschreibung der ebenfalls in Abbildung 38 dargestellten Pattern-Werkzeuge erfolgt in Kapitel 4.7.1.

4.6.2. Aufbau der Modelle

Die Beschreibung aller PaMGIS-Modelle umfasst einen eindeutigen Bezeichner für das Modell, seinen Typ, den Namen und eine textuelle Beschreibung des Modells. Darüber hinaus können auch noch Anmerkungen, eine Liste mit Referenzen, Kommentare von Benutzern und Usability-Feedback hinzugefügt werden. Das Beschreibungselement für das Usability-Feedback ist derzeit noch gänzlich unstrukturiert, d.h. es kann beliebige Inhalte in beliebiger Form aufnehmen. Die vom jeweiligen Modelltyp abhängigen modellspezifischen Anteile werden im Beschreibungselement <ModelSubjectMatter> hinterlegt.

Die für alle PaMGIS-Modelle relevanten Beschreibungselemente sind in Abbildung 39 visualisiert. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.1 einzusehen.

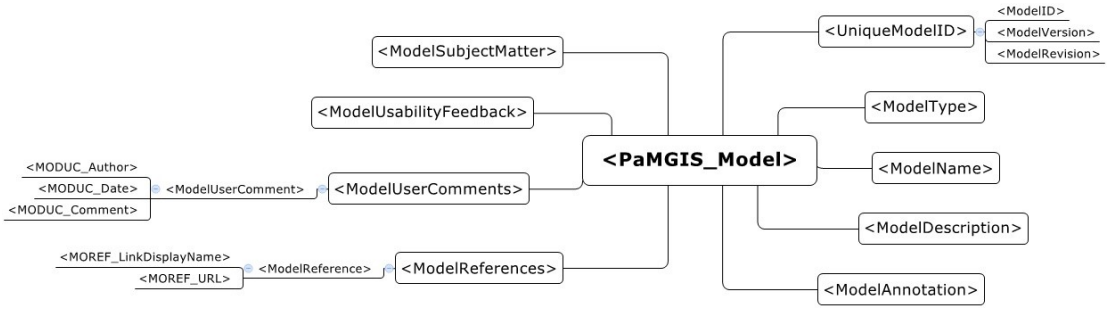


Abbildung 39 GENERELLER AUFBAU VON PAMGIS-MODELLEN <PAMGIS_MODEL>

In den folgenden Unterkapiteln wird der genaue Aufbau der verschiedenen PaMGIS-Modelle spezifiziert und erklärt.

4.6.2.1. Domänen-Modell

Das Domänen-Modell gehört zur höchsten Abstraktionsebene des PaMGIS-Frameworks (siehe Abbildung 16 in Kapitel 4.4.3). Es besteht aus zwei zwingend erforderlichen Teilmodellen, dem Aufgaben- und dem Konzept-Modell. Diese Teilmodelle stehen in sehr enger Beziehung zueinander und beinhalten unter anderem die Zuordnung von Konzepten zu Aufgaben und umgekehrt.

Neben den bei allen PaMGIS-Modellen vorhandenen, generellen Beschreibungselementen (siehe Kapitel 4.6.2) steht für die modellspezifischen Anteile des Domänen-Modells das Element <DomainModelReferences> zur Verfügung. Hier sind die Verweise auf die beiden Teilmodelle mittels der Elemente <TaskModelReference> und <ConceptModelReference>

unter Angabe des jeweiligen eindeutigen Bezeichners und Modellnamens sowie der betreffenden Versions- und Revisionsnummern untergebracht. Auf diese Weise kann zwischen verschiedenen Bearbeitungsständen der Modelle differenziert werden.

Alle relevanten Beschreibungselemente eines Domänen-Modells sind in Abbildung 40 zusammengefasst. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.2 enthalten.

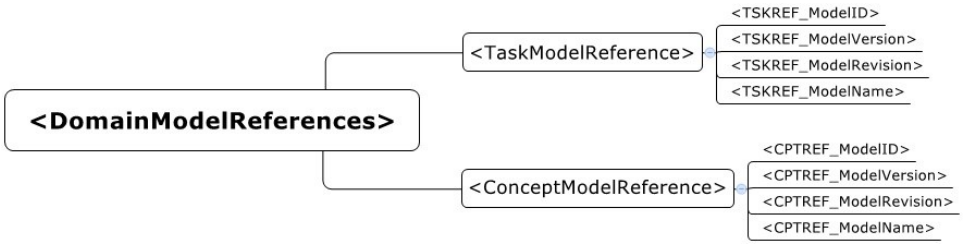


Abbildung 40 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES DOMÄNEN-MODELLS <DOMAINMODELREFERENCES>

4.6.2.1.1. Aufgaben-Modell

Ein Aufgaben-Modell dient im Kontext von PaMGIS zur Spezifikation aller Aufgaben und Teilaufgaben, die ein Benutzer zur Erreichung seiner Ziele durchführen muss (siehe Definition in Kapitel 2.2.1).

Die Spezifikation der eigentlichen Aufgaben und ihrer Beziehungen zueinander erfolgt auf Basis der *ConcurTaskTrees* (CTT) Notation, die auch bei den MB-UIDE *TERESA* und *MARIAE* zum Einsatz kommt. Hierbei werden die Aufgaben und die zu deren Erfüllung notwendigen Teilaufgaben in einer hierarchischen Baumstruktur angeordnet. Es erfolgt eine Unterscheidung von abstrakten, benutzerspezifischen, interaktiven und anwendungsspezifischen Task-Typen. Für die Modellierung der zeitlichen Abhängigkeiten zwischen den Aufgaben stehen sogenannte *Temporal Operators* zur Verfügung. Eine vollständige Übersicht über CTT befindet sich in [63].

Diese Methodik wurde für die Zwecke von PaMGIS derart erweitert, dass weitere Eigenschaften der Aufgaben und insbesondere die Beziehungen zwischen dem Aufgaben- und dem Konzept-Modell spezifiziert werden können. Beispielsweise kann, falls eine Aufgabe durch die Anwendung eines Musters in das Task-Modell eingefügt wurde, deren Herkunft mit dem Element <TaskOrigin> zu Zwecken der Rückverfolgung hinterlegt werden. Mit dem Beschreibungselement <ContextOfUse> kann angegeben werden, unter welchen Umständen, d.h. in welchen Benutzungskontexten die Aufgabe vorkommen soll. Darüber hinaus besteht die Möglichkeit, Vor- und Nachbedingungen für die Aufgaben zu definieren.

Alle spezifischen Beschreibungselemente des Aufgaben-Modells sind in Abbildung 41 aufgezeigt.

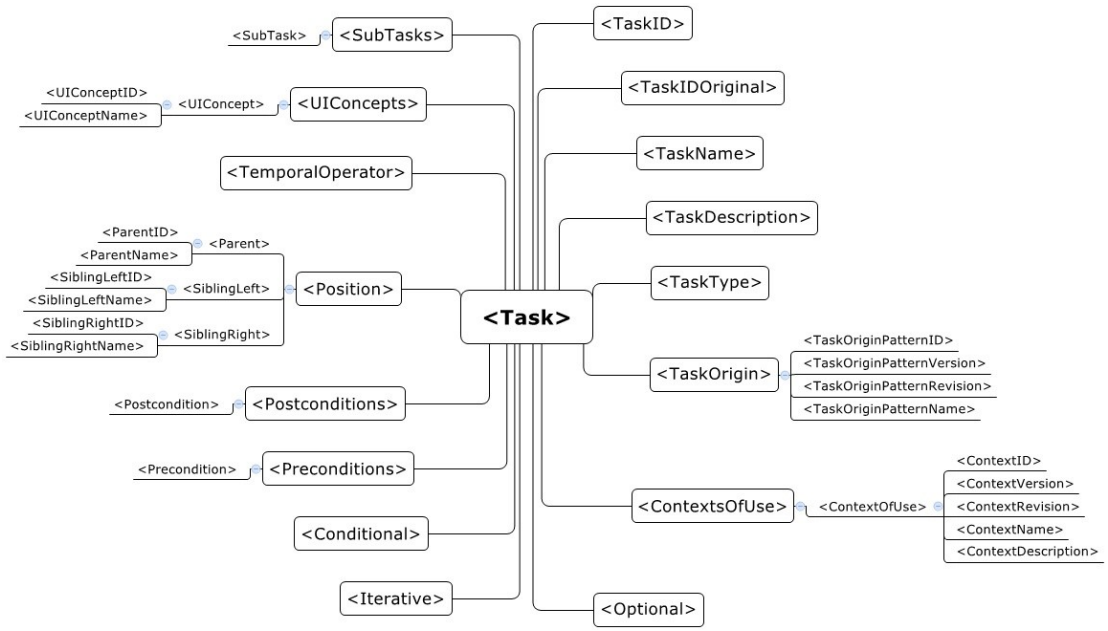


Abbildung 41 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES AUFGABEN-MODELLS <TASK>

Diese Struktur bildet auch die Basis für Beschreibung der Aufgaben-Modell-Fragmente, die den Mustern mithilfe des PPSL-Beschreibungselements <ModelFragment> zugeordnet werden können (siehe Kapitel 4.7.2.2.1).

Vor- und Nachbedingungen sind hierbei stets logische Ausdrücke und bestehen aus Operatoren und Operanden. Ihre Beschreibung erfolgt wie in Abbildung 42 gezeigt. Werden mehrere Vor- bzw. Nachbedingungen angegeben, so werden diese jeweils implizit durch einen logischen „UND“-Operator verknüpft.

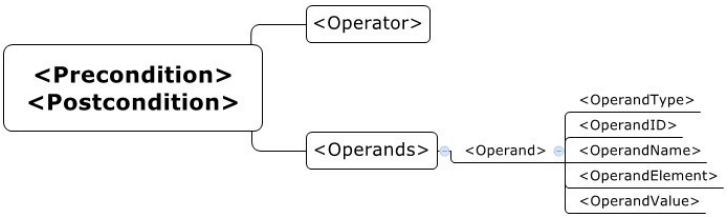


Abbildung 42 AUFBAU VON VOR- UND NACHBEDINGUNGEN <PRECONDITION>, <POSTCONDITION>

Eine vollständige Liste der Elemente zur Beschreibung des Aufgaben-Modells einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang E.2.1.

4.6.2.1.2. Konzept-Modell

Das Konzept-Modell dient im Kontext von PaMGIS zur Spezifikation aller, für die zukünftige Benutzeroberfläche relevanten Artefakte. Es kann praktisch auch als eine Art von Datenmodell für das User-Interface angesehen werden.

Das Konzept-Modell dient dazu, alle zur Benutzeroberfläche gehörenden Konzepte im Element <Concepts> aufzulisten. Jedes Konzept besitzt dabei einen eindeutigen Bezeichner,

einen Namen, einen Konzept-Typ, eine Beschreibung in Textform und einige weitere Eigenschaften, die, falls sie zum Zeitpunkt der Modellerstellung schon bekannt sind, gleich mit eingetragen werden können. Hierzu zählen beispielsweise die spätere Bezeichnung für das Konzept in der Benutzerschnittstelle (Label), der Datentyp, ein Vorbelegungswert, ob es sich um ein Pflichtfeld handelt, also zwingend ein Wert eingegeben werden muss und ob das Feld von Anfang an sichtbar und/oder aktiviert ist. Wie beim Aufgaben-Modell können auch hier Informationen zur Herkunft hinterlegt werden, wenn das betreffende Konzept durch die Anwendung eines Musters ins Modell aufgenommen wurde. Zudem ist es möglich, Vor- und Nachbedingungen für die Konzepte festzulegen (siehe Abbildung 42 auf Seite 145). Das Beschreibungselement <CCPT_DataLink> ermöglicht, eine Verbindung zu einem Datenelement im Datenmodell der zugrunde liegenden Geschäftsanwendung herzustellen. Schließlich kann noch durch Verweise auf das Task-Modell gekennzeichnet werden, welche dort enthaltenen Aufgaben das betreffende Konzept benötigen.

Alle relevanten Elemente zur Beschreibung eines Konzept-Modells sind in Abbildung 43 ersichtlich.

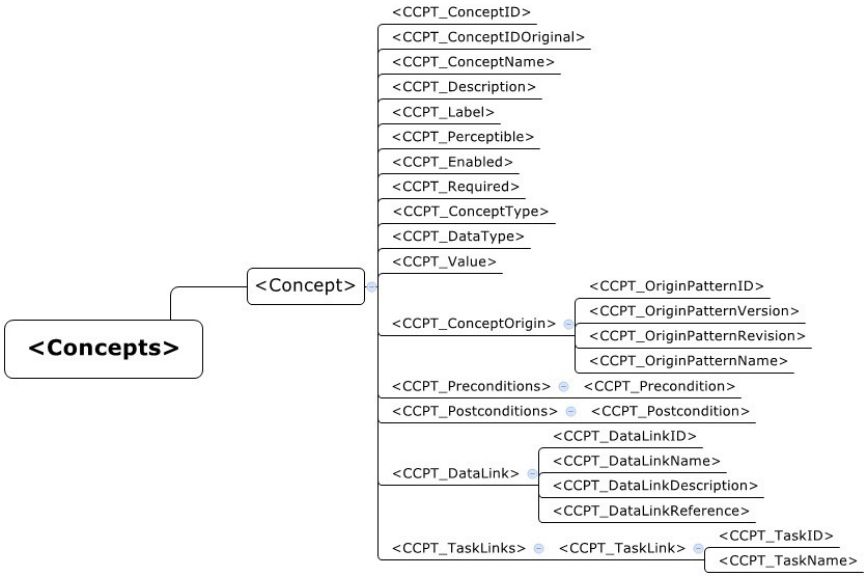


Abbildung 43 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONZEPT-MODELLS <CONCEPTS>

Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.2.2 enthalten.

Die von PaMGIS unterstützten Konzept-Typen sind in Tabelle 22 aufgelistet.

Tabelle 22 ÜBERSICHT ÜBER DIE DEFINIERTEN KONZEPT-TYPEN

Konzept-Typ	Beschreibung
DataInput	Zur Eingabe von Daten
DataOutput	Zur Ausgabe von Daten
DataEdit	Kombination von Ein- und Ausgabe zum Ändern von Daten
SingleChoice	Zur Auswahl eines Eintrags aus einer Menge von Einträgen
MultiChoice	Zur Auswahl eines oder mehrerer Einträge aus einer Menge von Einträgen

Konzept-Typ	Beschreibung
ChoiceItem	Einzelner Eintrag, der mit einem SingleChoice- oder MultiChoice-Element ausgewählt werden kann
Navigator	Zur Navigation, z. B. zwischen Dialogen
Activator	Zur Ausführung einer Routine, z. B. einer bestimmten Methode
UserFeedback	Zur Ausgabe von Rückmeldungen an den Benutzer, z. B. über den aktuellen Zustand des Systems
Alarm	Zur Ausgabe von Rückmeldungen an den Benutzer, z. B. in Fehlerfällen

4.6.2.2. Dialog-Modell

Mit dem Dialog-Modell werden die dynamischen Aspekte der zukünftigen Benutzerschnittstelle spezifiziert, indem auf Basis des Aufgaben-Modells und in Abhängigkeit des festgelegten Benutzungskontextes einzelne Dialoge definiert und dann deren Ablaufreihenfolge festgelegt wird. Dieses Modell wird der Kontext-Ebene zugeordnet, weil sein Inhalt sehr stark von den Informationen im Benutzungskontext-Modell abhängt. Zudem beeinflussen beide Modelle die späteren Transformationen der UI-Modelle gleichermaßen.

Die modellspezifischen Beschreibungen sind im Element <DialogCharacteristics> zusammengefasst. Mittels der Elemente <DomainModelReferences> und <ContextModelReference> werden die als Input benötigten Domänen- und Kontext-Modelle referenziert. Die eigentlichen Dialoge und ihre Ablaufreihenfolge werden im Element <Dialogs> auf Basis von Dialog-Graphen (siehe Kapitel 2.5.1.12) spezifiziert. Bei jedem Dialog wird angegeben, welche Aufgaben aus dem Task-Modell ihm zugeordnet werden. Dabei legt das Element <DLG_Processing> fest, ob nur die angegebene Aufgabe selbst oder der komplette, durch diese Aufgabe beginnende, CTT-Teilbaum in den Dialog eingeschlossen werden soll.

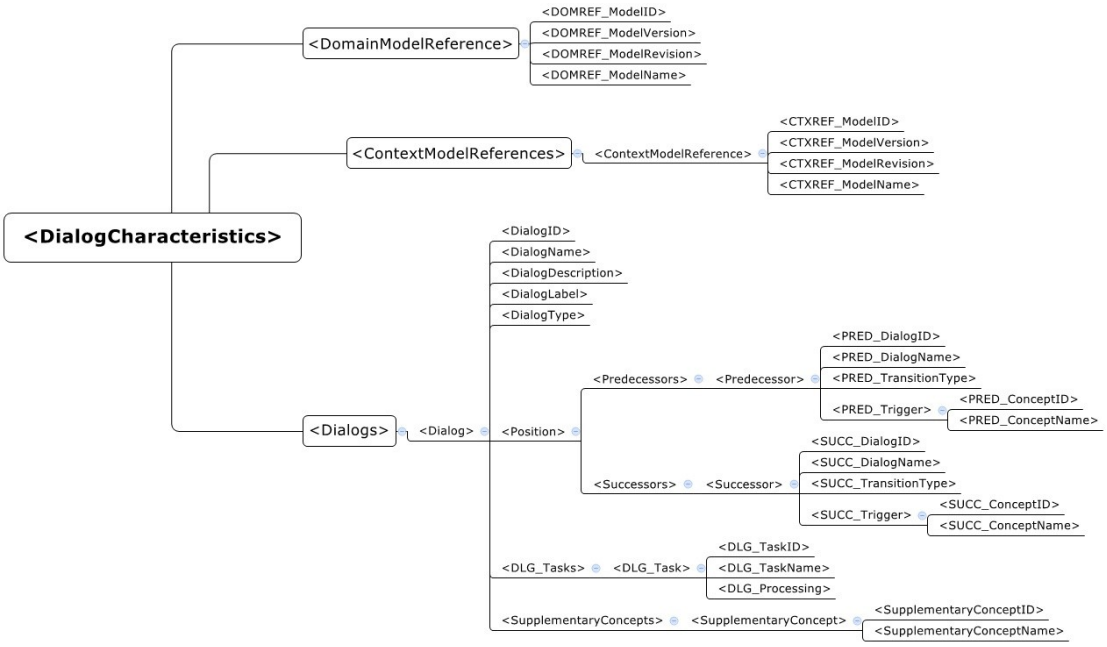


Abbildung 44 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES DIALOG-MODELLS <DIALOGCHARACTERISTICS>

Bei der Dialog-Modellierung kann es notwendig werden, zusätzliche Konzepte zur Navigation zwischen den Dialogen zu definieren. Diese Konzepte besitzen aber keine korrespondierenden Aufgaben im Task-Modell. Es ist daher nicht möglich, solche Konzepte über den herkömmlichen Weg, d. h. durch das Einfügen von Aufgaben, in den Dialog aufzunehmen. Dies wird mithilfe des Beschreibungselements <SupplementaryConcepts> erreicht, das dann eine oder mehrere Referenzen auf im Konzept-Modell enthaltene Konzepte beherbergt. Die Abfolge der Dialoge wird durch die Angabe entsprechender Vorgänger und Nachfolger mittels <Position> modelliert.

Alle relevanten Elemente zur Beschreibung eines Dialog-Modells sind in Abbildung 44 dargestellt. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.3 enthalten.

4.6.2.3. Benutzungskontext-Modell

Das Benutzungskontext-Modell, das abkürzend auch als Kontext-Modell bezeichnet wird, gehört hinsichtlich seines Abstraktionsgrades zur Kontext-Ebene des PaMGIS-Frameworks (siehe Abbildung 16 in Kapitel 4.4.3). Es besteht aus vier Teilmodellen, den Benutzer-, Geräte-, UI-Toolkit- und Umgebungs-Modellen. Hierbei müssen die Teilmodelle mit den Spezifikationen der Endgeräte und der verwendeten Programmierumgebung zwingend vorhanden sein, während die anderen beiden Modelle optional sind.

Wie alle anderen PaMGIS-Modelle umfasst auch die Spezifikation eines Kontext-Modells die generellen Beschreibungselemente der PaMGIS-Modelle (siehe Kapitel 4.6.2). Alle modellspezifischen Elemente, also die Verweise auf die Teilmodelle, sind in Abbildung 45 illustriert. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.4 enthalten.

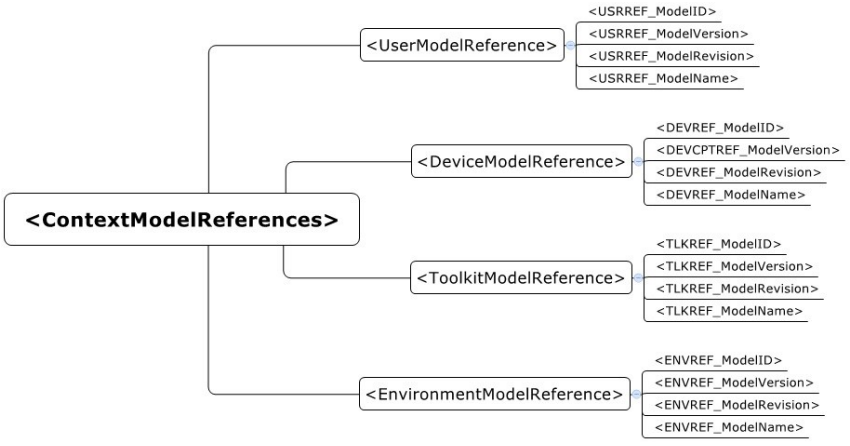


Abbildung 45 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONTEXT-MODELLS <CONTEXTMODELREFERENCES>

Die Modellierung des Benutzungskontexts kann also als Festlegung der relevanten Eigenschaften der Benutzer, des zum Einsatz kommenden Endgeräts, der verwendeten Programmierumgebung und der herrschenden Umwelteinflüsse aufgefasst werden. Ist in einem Kontext-Modell kein Benutzer- bzw. kein Umgebungs-Modell vorhanden, so wird davon ausgegangen, dass die Gestaltung der Benutzeroberfläche hiervon unabhängig ist, d.h. sie hat für alle Benutzer bzw. unter allen Umwelteinflüssen dasselbe Erscheinungsbild.

Die vier Teilmodelle werden in den nachfolgenden Unterkapiteln detailliert beschrieben. In der aktuellen Version bleiben die Modellierungsmöglichkeiten hierbei jeweils auf wesentliche Eigenschaften, die die Gestaltung der späteren Benutzeroberfläche signifikant beeinflussen können, beschränkt. Eine spätere Erweiterung der Modelle ist natürlich möglich und sinnvoll.

4.6.2.3.1. Benutzer-Modell

Im Benutzer-Modell werden wesentliche Charakteristika eines Users oder auch einer ganzen Gruppe von Benutzern definiert.

Die Eigenschaften der Benutzer sind im Element <UserCharacteristics> zusammengefasst. Hierzu zählen einerseits die körperlichen Fähigkeiten und andererseits die vorhandenen Erfahrungen bzw. das Wissen der User. Die Fähigkeiten beziehen sich auf die relevanten menschlichen Möglichkeiten, d.h. die Seh- und Hörfähigkeit und die mentalen und motorischen Fertigkeiten. Hierfür sind derzeit die Werte „hoch“ (engl. „high“), „mittel“ (engl. „medium“) und „gering“ (engl. „low“) vorgesehen. So kann beispielsweise hinsichtlich der Sehfähigkeit zwischen normal sehenden (hoch), sehbeeinträchtigten (mittel) und stark sehbehinderten bzw. gänzlich blinden Personen (gering) unterschieden werden. Die Fähigkeiten beeinflussen hauptsächlich die Art der möglichen Interaktion zwischen Mensch und Maschine, also praktisch die Wahl der Interaktionskanäle. Hinsichtlich der Erfahrungen werden momentan die vorhandenen Kenntnisse bezüglich der Anwendungsdomäne und die Erfahrungen im generellen Umgang mit elektronischen Geräten berücksichtigt. Hierbei kann jeweils zwischen „Experte“ (engl. „expert“), „Fortgeschrittener“ (engl. „advanced“) und „Novize“ (engl. „novice“) unterschieden werden. Die gewählten Fähigkeitsniveaus geben beispielsweise Hinweise darauf, ob und wieviel zusätzliche Informationen der Benutzer für die erfolgreiche Durchführung seiner Aufgaben benötigt. Darüber hinaus kann noch der Grad der Abgelenktheit des Users spezifiziert werden. Die möglichen Werte sind hierbei wiederum „hoch“, „mittel“ und „gering“. So kann z. B. bei einem Benutzer an seinem Büroarbeitsplatz eine hohe Konzentration, also ein geringer Grad der Abgelenktheit, unterstellt werden. Bei einem Autofahrer, der das Entertainment-System in seinem Fahrzeug während der Fahrt bedient, wäre demnach die Ablenkung aus Sicht des zu bedienenden Systems als hoch zu bezeichnen. Zudem enthalten die Benutzer-Eigenschaften noch die Geschäftsfähigkeit (engl. legal capacity) des betreffenden Users. Es kann dabei zwischen „voll“ (engl. „full“), „beschränkt“ (engl. „limited“) und „keine“ (engl. „none“) gewählt werden. Dieses Kriterium kann unter anderem signalisieren, dass verschiedene Funktionen der Benutzeroberfläche nur eingeschränkt oder gegebenenfalls gar nicht zur Verfügung stehen sollen, z. B. die Abgabe einer verbindlichen Bestellung in einer Anwendung für Online-Handel. Schließlich ist noch das Beschreibungselement <UserEmotionalState> vorgesehen, welches Informationen zum emotionalen Zustand aufnehmen und zu einem späteren Zeitpunkt weiter detailiert werden kann.

Sollte sich zu einem späteren Zeitpunkt herausstellen, dass die derzeit vorgegebenen Charakteristika und/oder deren Abstufungen, also die entsprechende Liste möglicher Werte, nicht ausreichend sind, so kann das Metamodell des Benutzermodells entsprechend verändert werden und darauf basierende, neue Modell-Versionen erstellt werden. Diese lassen sich dann jeweils anhand der Angaben zur Versions- und Revisionsnummer voneinander unterscheiden. Dies gilt gleichermaßen für alle Modelltypen des PaMGIS-Frameworks.

Alle spezifischen Elemente des Benutzer-Modells sind in Abbildung 46 zu sehen.

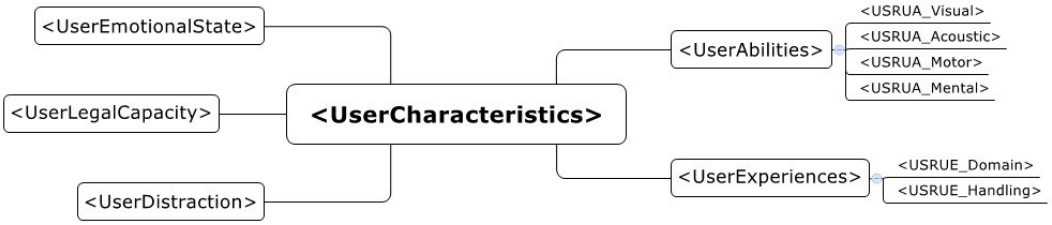


Abbildung 46 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES BENUTZER-MODELLS <USERCHARACTERISTICS>

Eine vollständige Liste der modellspezifischen Beschreibungselemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang E.4.1.

4.6.2.3.2. Geräte-Modell

Das Geräte-Modell beschreibt die für die Gestaltung der Benutzeroberfläche relevanten Eigenschaften des verwendeten Endgeräts.

Die Spezifikation der Charakteristika des Endgeräts erfolgt mithilfe des Elements <DeviceCharacteristics>. Dieses erlaubt, die für das betreffende Gerät zur Verfügung stehenden Betriebssysteme bzw. Betriebssystemversionen zu definieren. Ferner können die Möglichkeiten zur Ein- und Ausgabe festgelegt werden. Dabei wird wiederum zwischen den verschiedenen Interaktionskanälen unterschieden. Die Eingabe kann grundsätzlich motorisch, akustisch und/oder visuell erfolgen. Wenigstens einer dieser Kanäle muss explizit im Modell definiert sein. Zur motorischen Eingabe dienen eine Tastatur, ein berührungsempfindlicher Bildschirm und/oder ein Zeigegerät, wie z. B. eine Maus oder ein Trackball. Akustischer Input kann mittels Sprachsteuerung und visuelle Eingabe mithilfe von Gestensteuerung oder Eye-Tracking erfolgen. Jede dieser Möglichkeiten kann jeweils durch Zuweisung des Werts „ja“ (engl. „yes“) ausgewählt oder mit „nein“ (engl. „no“) abgewählt werden. Die Ausgabekanäle, bei denen ebenfalls wieder mindestens einer spezifiziert sein muss, adressieren jeweils den Seh-, Hör- und/oder den Tastsinn des Benutzers. Die visuelle Wahrnehmung erfolgt über einen Bildschirm (engl. screen), für den die Größe und die Auflösung definiert werden muss. Akustische Ausgaben erfolgen über zum Gerät gehörende Lautsprecher oder über einen Kopf- bzw. Ohrhörer während taktiler Output durch Vibrationen oder über ein Braille-Device realisiert wird. Für diese Funktionalitäten stehen wiederum die Booleschen Werte „ja“ und „nein“ zur Verfügung.

Bei der Zusammenstellung des Benutzungskontext-Modells kann aufgrund der ausgewählten Benutzer- und Device-Modelle eine erste Plausibilitätsprüfung erfolgen, ob das angedachte Endgerät für den betreffenden Benutzertyp geeignet ist.

Die spezifischen Elemente des Benutzer-Modells sind in Abbildung 47 aufgelistet.

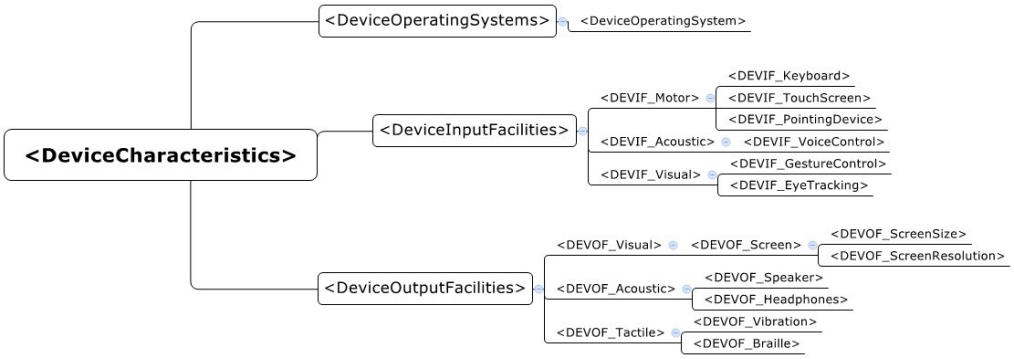


Abbildung 47 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES GERÄTE-MODELLS <DEVICECHARACTERISTICS>

Eine vollständige Liste der spezifischen Beschreibungselemente für das Geräte-Modell einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang E.4.2.

4.6.2.3.3. UI-Toolkit-Modell

Das UI-Toolkit-Modell, das abkürzend auch als Toolkit-Modell bezeichnet wird, beschreibt die durch die betreffende Programmiersprache bzw. -umgebung tatsächlich zur Verfügung stehenden UI-Objekte.

Die spezifischen Modellanteile sind im Beschreibungselement <ToolkitCharacteristics> zusammengefasst. Einerseits enthält dieses eine Auflistung derjenigen Betriebssysteme, für die das Toolkit verwendet werden kann. Andererseits verfügt es über eine Auflistung aller zur Verfügung stehenden UI-Objekte, für die jeweils der Objekttyp und eine Referenz zur jeweiligen Klassendefinition spezifiziert werden. Der Objekttyp wird bei der Transformation eines abstrakten in ein konkretes User-Interface für die entsprechende Zuordnung von UI-Objekten genutzt. Die Referenz ermöglicht das Auffinden der Klasse bei der Übersetzung der konkreten in die finale Benutzerschnittstelle (siehe Kapitel 4.6.3).

Bei der Zusammenstellung des Benutzungskontext-Modells kann auf Basis in den Teilmodellen enthaltenen Informationen überprüft werden, ob das UI-Toolkit auf dem ausgewählten Endgerät einsetzbar ist.

Die spezifischen Elemente des UI-Toolkit-Modells sind in Abbildung 48 dargestellt.

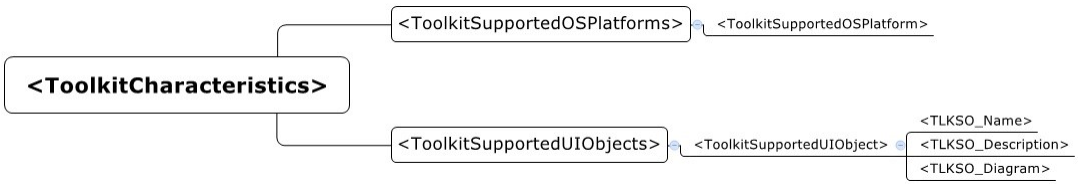


Abbildung 48 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES TOOLKIT-MODELLS <TOOLKITCHARACTERISTICS>

Eine vollständige Liste der spezifischen Beschreibungselemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang E.4.3.

4.6.2.3.4. Umgebungs-Modell

Anhand des Umgebungs-Modells werden Umwelteinflüsse, die bei der Gestaltung der Benutzerschnittstelle berücksichtigt werden sollen, definiert.

Die eigentlichen Umgebungseigenschaften werden mithilfe des Elements <EnvironmentCharacteristics> spezifiziert. Derzeit können Angaben zu möglichen Beeinträchtigungen durch Lichtreflexion (engl. luminous reflectance), Umgebungsgeräusche (engl. ambient noise) und Staub- oder Schmutzbelastung (engl. pollutant) gemacht werden. Lichtreflexion kann beispielsweise bei einer Verwendung im Außenbereich durch die Lichteinstrahlung der tief stehenden Sonne oder durch grelle Beleuchtungseinrichtungen entstehen und somit den Einsatz des visuellen Interaktionskanals infrage stellen. Analog gilt dies für den Einsatz von akustischer Ein- und/oder Ausgabe bei hohem Geräuschpegel. Bei einer Belastung durch Schmutz und Staub in der Luft oder auf dem Endgerät kann einerseits die Sicht des Benutzers oder die Funktionalität des Geräts beeinträchtigt sein. Andererseits ist es auch möglich, dass der Anwender Schutzkleidung, wie Handschuhe, Helm, Schutzbrille oder Schutzanzug trägt und dadurch die Sicht, das Hörvermögen und/oder die Beweglichkeit eingeschränkt werden. Dies wäre beispielsweise für Systeme relevant, die zur Verwendung durch Angehörige der Feuerwehr bei Einsätzen in stark verrauchten Räumlichkeiten gedacht sind.

Eine Übersicht über die spezifischen Elemente des Umgebungs-Modells wird in Abbildung 49 gegeben.

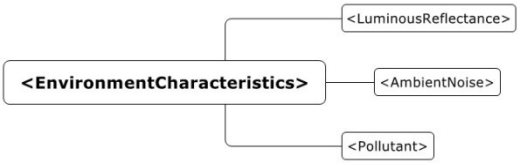


Abbildung 49 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES UMGEBUNGS-MODELLS <ENVIRONMENTCHARACTERISTICS>

Eine vollständige Liste der spezifischen Beschreibungselemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang E.4.4.

4.6.2.4. Benutzeroberflächenmodelle

Die bisher beschriebenen Modelle sind, wie in Abbildung 16 in Kapitel 4.4.3 dargestellt, der Domänen- und der Benutzungskontext-Ebene des PaMGIS-Frameworks zuzuordnen und besitzen somit ein hohes Abstraktionsniveau. Auf den darunter liegenden Ebenen nimmt der Abstraktionsgrad sukzessive ab, d. h. die Beschreibung der Benutzerschnittstelle wird immer konkreter. Es handelt sich dabei in genau dieser Reihenfolge um die Ebenen der abstrakten (AUI), konkreten (CUI) und finalen (FUI) Benutzerschnittstellen. Die betreffenden Modelle werden in den folgenden Unterkapiteln näher betrachtet.

4.6.2.4.1. AUI-Modell

Das abstrakte User-Interface-Modell ist die abstrakteste Repräsentation der angestrebten Benutzeroberfläche. Es definiert die zugehörenden, abstrakten UI-Elemente und fasst diese in strukturierter Art und Weise zusammen.

Die Beschreibung von AUI-Elementen ist der von Konzepten (siehe Kapitel 4.6.2.1.2) sehr ähnlich. Sie besitzen jeweils einen eindeutigen Bezeichner, einen Namen, einen Elementtyp, eine Beschreibung in Textform und einige weitere Eigenschaften, die, falls sie zum Zeitpunkt der Modellerstellung schon bekannt sind, erfasst werden können. Hierzu zählen die spätere Bezeichnung für das Konzept in der Benutzerschnittstelle (Label), der Datentyp, ein etwaiger Vorbelegungswert, ob es sich um ein Pflichtfeld handelt und ob das Feld von Anfang an für den User sichtbar und aktiv nutzbar ist. Zudem sind Informationen zur Herkunft der AUI-Elemente hinterlegt, d. h. es enthält Verweise auf die Konzepte, aufgrund derer das betreffende AUI-Element erzeugt wurde. Analog zu den Konzepten können Vor- und Nachbedingungen definiert und mithilfe des Beschreibungselements <AUIE_DataLink> eine Verbindung zu einem Datenelement im Datenmodell der zugrunde liegenden Geschäftsanwendung hergestellt werden.

Darüber hinaus kann für sogenannte Container-Elemente durch das Beschreibungselement <AUIE_ContainedAUIElements> festgelegt werden, welche anderen AUI-Elemente diese beinhalten sollen.

Alle relevanten Elemente zur Beschreibung eines AUI-Modells sind in Abbildung 50 illustriert. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.5 nachzulesen.

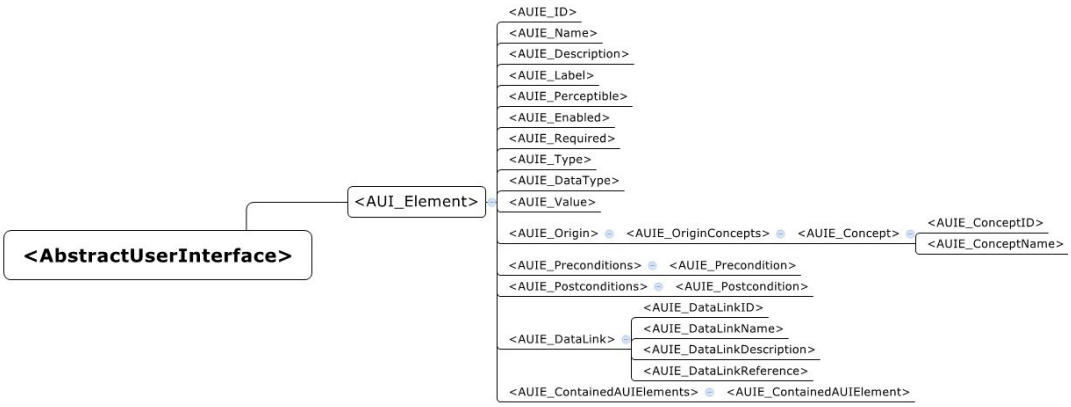


Abbildung 50 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES ABSTRAKTEN UI-MODELLS <ABSTRACTUSERINTERFACE>

Die von PaMGIS unterstützten Typen von AUI-Elementen sind in Tabelle 23 aufgelistet. Grundsätzlich gibt es zu jedem Konzept-Typ einen korrespondierenden AUI-Element-Typ. Hinzu kommen die bereits erwähnten Container-Elemente, die zur Strukturierung der Benutzerschnittstelle beitragen. Dies sind die Typen „Cluster“ und „Grouping“.

Tabelle 23 ÜBERSICHT ÜBER DIE DEFINIERTEN TYPEN VON AUI-ELEMENTEN

AUI-Element-Typ	Beschreibung
DataInput	Zur Eingabe von Daten
DataOutput	Zur Ausgabe von Daten
DataEdit	Kombination von Ein- und Ausgabe zum Ändern von Daten
SingleChoice	Zur Auswahl eines Eintrags aus einer Menge von Einträgen
MultiChoice	Zur Auswahl eines oder mehrerer Einträge aus einer Menge von Einträgen
ChoiceItem	Einzelner Eintrag, der mit einem SingleChoice- oder MultiChoice-Element ausgewählt werden kann
Navigator	Zur Navigation, z. B. zwischen Dialogen
Activator	Zur Ausführung einer Routine, z. B. einer bestimmten Methode
UserFeedback	Zur Ausgabe von Rückmeldungen an den Benutzer, z. B. über den aktuellen Zustand des Systems
Alarm	Zur Ausgabe von Rückmeldungen an den Benutzer, z. B. in Fehlerfällen
Cluster	Container-Element der höchsten Ordnung, das AUI-Elemente beliebigen Typs beinhalten kann
Grouping	Container-Element, das bis auf Cluster-Elemente AUI-Elemente beliebigen Typs beinhalten kann

4.6.2.4.2. CUI-Modell

Die nächste Konkretisierungsstufe nach dem AUI ist die konkrete Benutzerschnittstelle. Die Beschreibungsstruktur entspricht im Wesentlichen der des AUI-Modells. Es sieht jedoch zusätzliche Beschreibungselemente, die das spätere Erscheinungsbild der UI-Objekte spezifizieren, vor. Dies wird durch das Beschreibungselement <CUIE_Presentation> bewerkstelligt, das Informationen über Größe, Position, Ausrichtung, Farbe und Rahmen der Artefakte aufnehmen kann. Die Information zur Herkunft stellt das Element <CUIE_Origin> bereit. Es enthält Verweise auf diejenigen AUI-Elemente, aufgrund derer das betreffende CUI-Element erzeugt wurde.

Die Elemente zur Beschreibung eines CUI-Modells sind in Abbildung 51 dargestellt. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten sind in Anhang E.6 zu finden.

Die möglichen Typen von AUI-Elementen sind in Tabelle 24 aufgelistet. Wie beim AUI sind auch hier Container-Elemente verfügbar, die zur Strukturierung der Benutzerschnittstelle beitragen. Dies sind die Typen „Window“, „Form und „Box“.

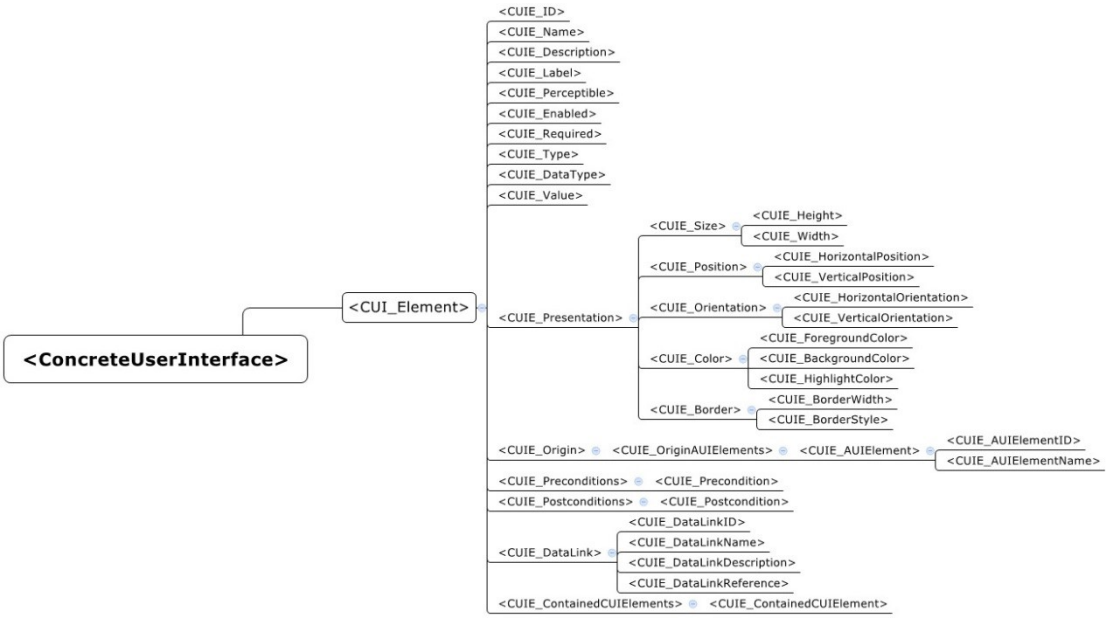


Abbildung 51 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONKRETEN UI-MODELLS <CONCRETEUSERINTERFACE>

Tabelle 24 ÜBERSICHT ÜBER DIE DEFINIERTEN TYPEN VON CUI-ELEMENTEN

CUI-Element-Typ	Beschreibung
InputField	Feld zur Eingabe von Daten
OutputField	Feld zur Ausgabe von Daten
EditField	Feld zur Ein- und Ausgabe bzw. zum Ändern von Daten
Menu	Auswahl-Menü
MenuEntry	Einzelne auswählbare Option in einem Auswahl-Menü
DropdownList	Sich öffnende Liste mit Auswahloptionen, die sich nach erfolgter Auswahl wieder schließt und nur noch das ausgewählte Element anzeigt
DropdownListEntry	Einzelne auswählbare Option in einer DropdownList
Listbox	Ähnlich wie eine Dropdownliste, nur sind stets mehrere Auswahloptionen sichtbar, wobei ausgewählte Einträge farblich markiert sind
ListboxEntry	Einzelne auswählbare Option in einer Listbox
RadioButtons	Gruppe von mehreren RadioButton-Elementen, bei der im Falle der Auswahl einer Option die anderen abgewählt werden
RadioButton	Repräsentiert eine der in einer Gruppe von RadioButtons zur Auswahl stehenden Optionen
CheckBox	Auswahl-Box zur binären Selektion, z.B. zur An- bzw. Abwahl einer Option
Button	Schaltfläche zum Auslösen von Aktionen
Window	Fenster, das andere Elemente beinhalten kann (ausser Window)
Form	Formular, das andere Elemente beinhalten kann
Box	Rahmen, der andere Elemente umschliessen kann

4.6.2.4.3. FUI-Modell

Die finale Benutzerschnittstelle besteht aus dem Quell-Code der im Kontext-Modell spezifizierten Zielplattform. Dieser wird entweder mithilfe eines Interpreters ausgeführt oder mit einem Compiler in ausführbaren Maschinen-Code übersetzt.

Naturgemäß hängt die Beschreibungsstruktur des FUI-Modells stark von der eingesetzten Programmiersprache und dem verwendeten UI-Toolkit ab. Die zur Verfügung stehenden UI-Elemente sind im jeweiligen UI-Toolkit-Modell enthalten.

Bei der im Rahmen dieser Dissertation durchgeführten Fallstudie wurde die Programmiersprache *Python* in Verbindung mit dem hierfür angepassten GUI-Toolkit *Tkinter* eingesetzt. Das gesamte Toolkit-Modell mit allen einsetzbaren UI-Elementen kann in Anhang K.6 eingesehen werden.

4.6.3. Modell-Transformationen

Die Ableitung der ablauffähigen Benutzerschnittstelle aus dem Domänen-Modell erfolgt bei PaMGIS in drei Phasen. Dabei wird zunächst aus dem Konzept-Modell das AUI-Modell generiert, aus dem anschließend das CUI-Modell konstruiert wird. Zum Abschluss wird das CUI- dann in das FUI-Modell transformiert.

Es lassen sich hierbei also folgende drei Modell-Transformation unterscheiden:

1. Das Domänen- bzw. Konzept-Modell in das AUI-Modell (DOM2AUI)
2. Das AUI- in das CUI-Modell (AUI2CUI)
3. Das CUI- in das FUI-Modell (CUI2FUI)

Als Hilfsmittel für diese Transformationen kommen Zuordnungstabellen, die auch als Mapping-Tabellen bezeichnet werden, zum Einsatz.

Wie die Modelle, werden auch die Tabellen in PaMGIS mit einer XML-konformen Notation spezifiziert. Tabellen-Beschreibungen umfassen einen eindeutigen Bezeichner für die Tabelle, den Typ, einen Namen sowie eine textuelle Beschreibung und Anmerkungen. Darüber hinaus können auch noch Kommentare von PaMGIS-Benutzern hinzugefügt werden. Das Kernstück bilden die Beschreibungselemente <ApplicableContextsOfUse> und <MappingRules>. Ersteres beinhaltet Referenzen auf diejenigen Kontext-Modelle, für die Zuordnungstabelle einsetzbar ist. Letzteres beherbergt die eigentlichen Zuordnungsregeln und besteht aus Unterelementen für eine laufende Regelnummer, den Typ des zu transformierenden UI-Elements, den Typ des Ziel-Elements sowie eine Bedingung, die erfüllt sein muss, damit die Transformation tatsächlich durchgeführt wird. Zudem können die Regeln noch mit textuellen Anmerkungen versehen werden.

Die für alle PaMGIS-Tabellen relevanten Beschreibungselemente sind in Abbildung 52 enthalten. Weitergehende Informationen mit Erklärungen sowie Angaben zu Datentypen und Kardinalitäten können in Anhang F.1 eingesehen werden.

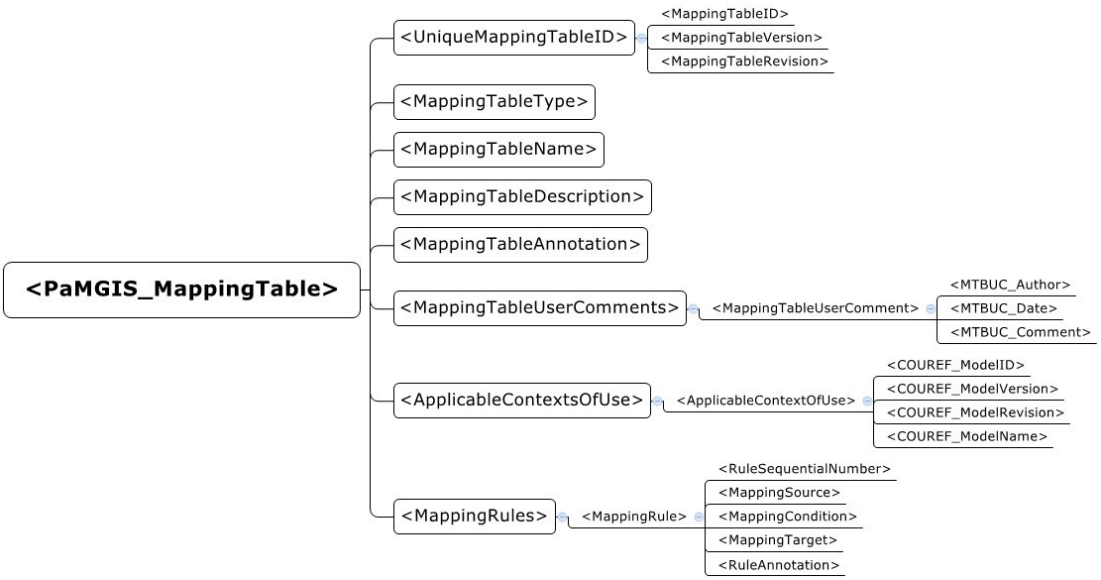


Abbildung 52 BESCHREIBUNGSELEMENTE DER ZUORDNUNGSTABELLEN FÜR MODELLTRANSFORMATIONEN <PAMGIS_MAPPINGTABLE>

4.6.3.1. Domänen- zu AUI-Modell

Das abstrakte User-Interface ist bei PaMGIS unabhängig von jeglichem Benutzungskontext, d. h. es handelt sich für alle Benutzer, Endgeräte, Toolkits und Umgebungsbedingungen um genau dasselbe AUI.

Da dies auch für das Konzept-Modell gilt, wird für jedes dort enthaltene Konzept ein entsprechendes AUI-Element gleichen Typs angelegt und dem AUI-Modell hinzugefügt. Dabei werden bis auf zwei Ausnahmen die in den Konzept-Beschreibungen enthaltenen Informationen übernommen. Ausgenommen hiervon ist einerseits der eindeutige Bezeichner <AUIE_ID>, der bei der Erzeugung des AUI-Elements neu vergeben wird. Andererseits betrifft dies das Beschreibungselement mit der Herkunftsinformation <AUIE_Origin>, in dem der eindeutige Bezeichner und der Name des betreffenden Konzepts vermerkt werden.

Die Zuordnungstabelle für diese Transformation enthält in der Regel alle Konzept-Typen als Quell-Elemente und die entsprechenden – namensgleichen – Typen von AUI-Elementen als Ziele. Das Beschreibungselement <ApplicableContextsOfUse> fehlt oder bleibt leer. Somit wird indiziert, dass die Tabelle für alle möglichen Benutzungskontexte einsetzbar ist.

4.6.3.2. AUI- zu CUI-Modell

In diesem Schritt erfolgt eine erste Anpassung des Benutzeroberflächenmodells an den zuvor spezifizierten Benutzungskontext. Für diese Adaption wird zunächst das AUI-Modell modifiziert. Hierbei sind folgende Aktionen auszuführen:

1. AUI-Elemente, die im gewünschten Benutzungskontext nicht benötigt werden, werden aus dem Modell entfernt.
2. Zusätzliche AUI-Elemente, die beispielsweise zur Strukturierung der Benutzerschnittstelle dienen oder für Navigation benötigt werden, werden in das Modell integriert.

3. Das Modell wird durch das Einfügen von Container-Elementen strukturiert.

Alle hierfür benötigten Informationen sind im Dialog-Modell für den angestrebten Kontext und im unterliegenden Task-Modell enthalten.

Für jeden im Dialog-Modell definierten Dialog wird ein Container-Element in das AUI-Modell eingefügt. Es handelt sich dabei in der Regel um ein <Cluster>-Element. In dieses werden dann alle zu den im Dialog zusammengefassten Aufgaben zugehörenden AUI-Elemente verschoben. Sie befinden sich dann im Beschreibungselement <AUIE_ContainedAUIElements> des Clusters.

Enthält der Dialog Aufgaben, denen kein Konzept zugeordnet ist, so werden für diese an der entsprechenden Stelle <Grouping>-Elemente im AUI-Modell eingefügt. In diese Art von Container werden alle AUI-Konzepte verschoben, die zu den betreffenden Unteraufgaben, d. h. zu den im Unterbaum des Task-Modells auftretenden Aufgaben, gehören.

Falls im Dialog zusätzliche Konzepte spezifiziert sind, die bei der Dialogmodellierung zu Zwecken der Navigation eingefügt wurden, so werden für diese korrespondierende AUI-Elemente im AUI-Modell angelegt.

Sobald die Kontextanpassung abgeschlossen ist, kann das adaptierte AUI-Modell in ein CUI-Modell transformiert werden. Basis hierfür ist die für den Benutzungskontext definierte Zuordnungstabelle mit den entsprechenden Mapping-Regeln.

Ein praktisches Durchführungsbeispiel für die Konstruktion und kontextspezifische Modifikation eines AUI-Modells und dessen Überführung in ein CUI-Modell mithilfe einer Mapping-Tabelle des Typs AUI2CUI wurde im Rahmen der Fallstudie erstellt und kann in Kapitel 5.6.2 eingesehen werden.

4.6.3.3. CUI- zu FUI-Modell

Schließlich wird noch das CUI- in ein FUI-Modell, das die fertige Benutzeroberfläche darstellt, überführt. Auch für diese Transformation wird eine entsprechende Zuordnungstabelle benötigt. Mit ihr wird die automatisierte, zielplattformspezifische Generierung des Programmcodes gesteuert.

Für die Fallstudie erfolgte die Transformation und prototypische Implementierung der finalen Benutzerschnittstelle mit der Programmiersprache *Python* und dem hierfür angepassten UI-Toolkit *Tkinter*. Details hierzu können Kapitel 5.6.3 entnommen werden.

4.7. Detailaspekte des musterbasierten Anteils

In diesem Kapitel wird zunächst eine Übersicht über die benötigten Werkzeuge des musterbasierten Teils von PaMGIS gegeben. Anschließend wird der Aufbau der Pattern-Beschreibungssprache PPSL im Detail erläutert.

4.7.1. Übersicht über die Pattern-Werkzeuge

Eine visuelle Übersicht über die Werkzeuge für das Pattern-Repository und das Modell-Repository wurde bereits in Abbildung 38 dargestellt (siehe Kapitel 4.6.1).

Analog zur entsprechenden Funktion beim modellgetriebenen Teil von PaMGIS wird das Werkzeug *Pattern Meta Modell Administration* von der Rolle des Framework-Adminstrators genutzt. Er kann damit das bestehende ontologische Modell der Pattern-Beschreibungssprache, das ebenfalls einer Versionierung unterliegt, editieren. Mit der Freigabe steht die neue Version dann für die weitere Verwendung zur Verfügung. Aus Konsistenzgründen werden keine Versionen von PPSL gelöscht. Sie stehen aber nach ihrer Zurückziehung für eine neuerliche Verwendung nicht mehr bereit.

Das Werkzeug *Pattern & Pattern Language Model Authoring* stellt vergleichbare Funktionalitäten wie die Authoring-Tools des Modell-Repository zur Verfügung. Es dient dazu, wie in Kapitel 4.4.1 beschrieben, neue archetypische bzw. benutzerdefinierte Muster anzulegen und bestehende Patterns zu verändern. Die archetypischen Patterns unterliegen, wie auch PPSL, einer Versionierung. Auch sie werden aus Konsistenzgründen nicht gelöscht, können aber durch den Pattern-Administrator zurückgezogen werden. Hingegen können benutzerdefinierte Muster sowohl gelöscht als auch ohne Anlegen einer neuen Version beliebig verändert werden. Darüber hinaus werden, wie schon bei den Modell-Werkzeugen beschrieben, Browsing und Suche im Pattern-Repository unterstützt. Durch die Spezifikation von Beziehungen zwischen den Mustern besteht die Möglichkeit, aus einer Menge einzelner Patterns eine Pattern-Sprache zu definieren.

Mit der Funktion *Pattern & Pattern Language Dissemination* wird der Web-basierte Zugriff für anonyme Benutzer realisiert. Ihnen werden Möglichkeiten zum Browsing und für die Suche von eigens dafür freigegebenen Mustern im Pattern-Repository zur Verfügung gestellt. Eine weitere Einschränkung hierbei ist, dass nicht alle PPSL-Beschreibungselemente der Pattern-Spezifikationen auf diesem Weg einsehbar sind.

Die Werkzeuge *Domain Model Authoring* und *Dialog Model Authoring* erlauben es, Modell-Fragmente zu erstellen und den Pattern-Spezifikationen beizufügen. Modell-Fragmente sind Teilmodelle, die bei der Erstellung der entsprechenden Modelle im modellgetriebenen Teil von PaMGIS wie Bausteine verwendet werden können. Die Fragmente können neben den eigentlichen Modellanteilen zusätzlich noch Hinweise über deren Integration in das jeweilige Zielmodell enthalten (siehe Kapitel 4.7.2.2).

Alle weiteren, im mittleren Teil von Abbildung 38 dargestellten Funktionalitäten wurden bereits in Kapitel 4.6.1 beschrieben und sind in analoger Weise für den musterbasierten Teil von PaMGIS verwendbar.

4.7.2. Beschreibungssprache für Muster

Das mit der Pattern-Beschreibungssprache des PaMGIS-Frameworks (PPSL) verfolgte Ziel ist, eine *Lingua franca* in zweifacher Hinsicht zu schaffen. Einerseits soll eine Überführung von bereits bestehenden, aber durchaus unterschiedlichen Beschreibungssprachen (siehe Kapitel 3.2.1) in das PPSL-Format mit möglichst geringem Informationsverlust erfolgen können. Andererseits soll durch die Anreicherung der Pattern-Spezifikationen um Modell-Informationen die Verbindung von musterbasierten und modellgetriebenen Entwicklungsansätzen erreicht werden (siehe Abbildung 14 in Kapitel 4.1.2).

4.7.2.1. Aufbau der PaMGIS Pattern Specification Language

Im Folgenden werden die Beschreibungselemente von PPSL, die im weiteren Verlauf auch abkürzend nur als Elemente bezeichnet werden, erläutert. Es werden dabei die englischen Bezeichnungen verwendet, die durch spitze Klammern gekennzeichnet sind.

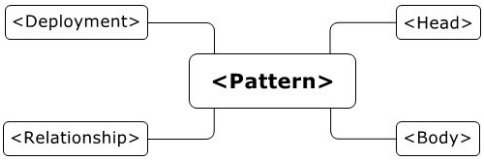


Abbildung 53 GLIEDERUNG VON PPSL DURCH DIE VIER TOP-LEVEL-ELEMENTE

Auf der obersten Ebene werden, wie in Abbildung 53 gezeigt, die Pattern-Spezifikationen durch die vier Elemente <Head>, <Body>, <Relationship> und <Deployment> strukturiert. Hierbei enthält <Head> die Metadaten zum betreffenden Pattern und <Body> die eigentliche Beschreibung des Musters. Des Weiteren werden in <Relations> die bestehenden Beziehungen zu anderen Patterns definiert und <Deployment> enthält Informationen hinsichtlich der tatsächlichen Implementierung des Musters wie beispielsweise Modell-Fragmente, die im modellgetriebenen Teil von PaMGIS verwendet werden können.

Das <Head>-Element umfasst Informationen hinsichlich der für die betreffende Pattern-Spezifikation verwendeten PPSL-Version, einen eindeutigen und aus mehrerer Unterelementen bestehenden Bezeichner und Angaben zur Klassifikation des Musters. Zudem besteht die Möglichkeit, einen Name und weitere, alternative Bezeichnungen, Angaben zum Bearbeitungsstatus, eine prägnante Zusammenfassung der Inhalte (Synopsis) und Angaben zu den Personen, die zur vorliegenden Pattern-Spezifikation beigetragen haben, anzugeben. Darüber hinaus können Angaben zur Historie des Musters hinterlegt werden. Hierfür können einerseits, falls ein existierendes Muster aus einer anderen Pattern-Sammlung übernommen wurde, Informationen zu dessen Herkunft und andererseits auch die eventuell vorgenommenen Änderungen in ihrer zeitlichen Abfolge hinzugefügt werden.

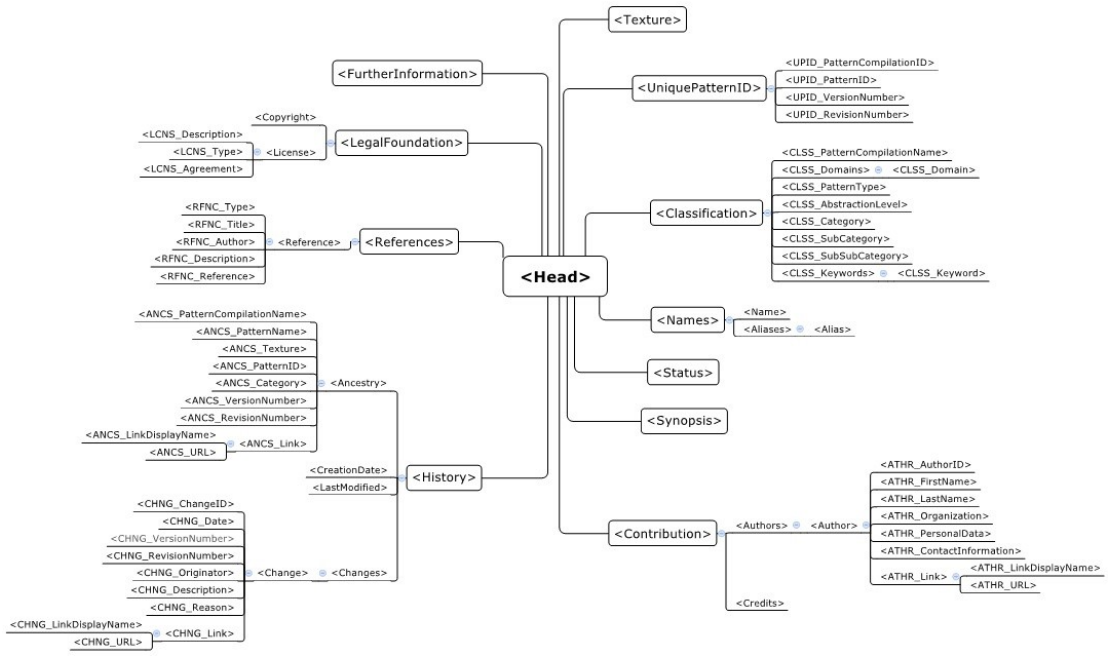


Abbildung 54 IM TOP-LEVEL-ELEMENT <HEAD> ZUSAMMENGEFASSTE PPSL-BESCHREIBUNGSELEMENTE

Ferner bietet das <Head>-Element Möglichkeiten, Hinweise sowohl auf Quellen und weiterführende Dokumente als auch zu rechtlichen Aspekten, wie beispielsweise zum Urheberrecht zu integrieren. Schließlich können dem Unterelement <FurtherInformation> noch beliebige anderweitige Informationen, die in keinem anderen Beschreibungselement untergebracht werden können, zugeordnet werden.

Die Metadaten werden unter anderem beim Browsen und bei der Suche im Pattern-Repository ausgewertet. Eine Übersicht über alle Unterelemente des Beschreibungselements <Head> können aus Abbildung 54 entnommen werden.

Das <Body>-Element repräsentiert die eigentliche Beschreibung des Musters und ist in die zwei Unterelemente <Theory> und <Practice> eingeteilt. Ersteres beherbergt die von Christoph Alexander definierte dreiteilige Regel in Form von Problem-, Kontext- und Lösungsbeschreibungen, die jeweils in einer kurzen und/oder ausführlichen Fassung angegeben werden können. Ferner können Informationen zu den auf das Pattern einwirkende Kräfte (engl. forces), Begründungen und Einschätzung der Wahrscheinlichkeit für die Korrektheit der Lösung, Beschreibungen des sich nach der Anwendung des Patterns ergebenden Kontextes sowie Diagramme und Schaubilder hinterlegt werden. Das Unterelement <Practice> nimmt Beispiele (engl. examples) und sogenannte Known-Uses auf. Diese unterscheiden sich dadurch, dass sich die Beispiele auf Realisierungen mithilfe von PaMGIS beziehen und jeweils auf Modell-Fragmente innerhalb des Beschreibungselements <Deployment> verwiesen wird. Known-Uses hingegen beziehen sich auf beliebige Implementierungen ausserhalb von PaMGIS, bei denen beispielsweise ein *Uniform Resource Locator* (URL) der entsprechenden Realisierung enthalten ist.

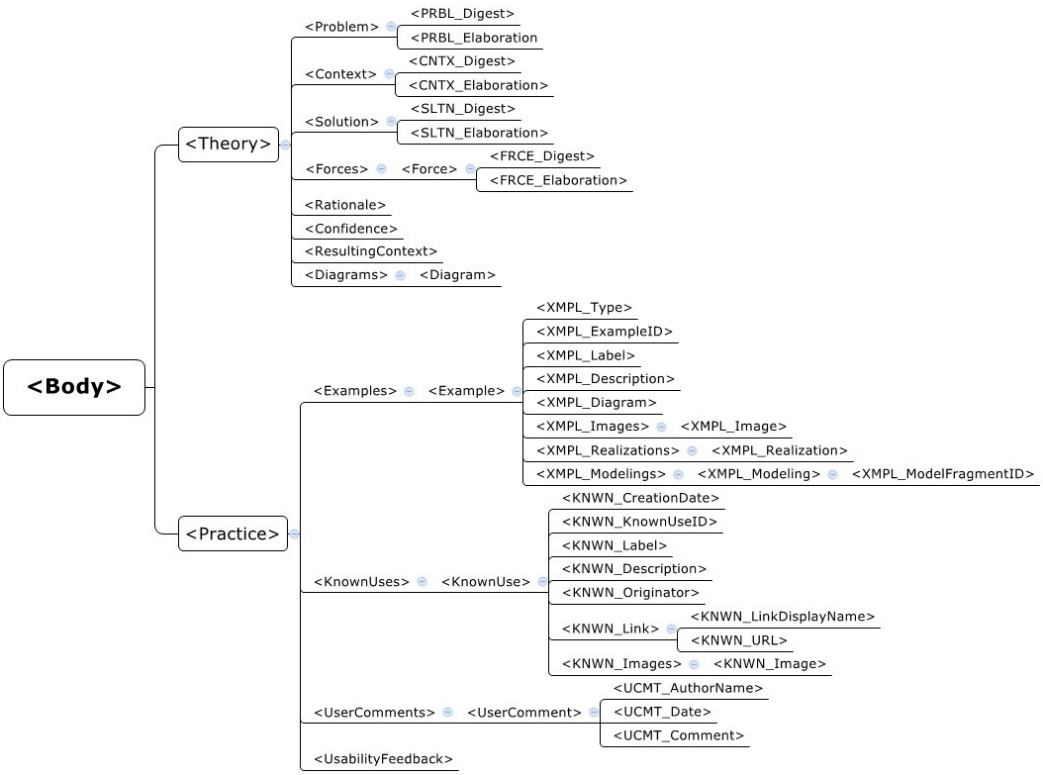


Abbildung 55 IM TOP-LEVEL-ELEMENT <BODY> ZUSAMMENGEFASSTE PPSL-BESCHREIBUNGSELEMENTE

Darüber hinaus können in diesem Teil auch noch Kommentare von Benutzern und Usability-Feedback abgelegt werden. Hierbei ist das Element <UsabilityFeedback> derzeit noch gänzlich unstrukturiert, d.h. es kann beliebige Inhalte in beliebiger Form aufnehmen.

Die Beschreibungen aller Unterelemente des Elements <Body> sind in Abbildung 55 zusammengefasst.

Mithilfe des <Relationship>-Elements können Beziehungen zu anderen Mustern definiert werden. Dies ist ein grundlegendes Mittel um Pattern-Sprachen zu erstellen. Einerseits können die Relationen in Textform beschrieben werden, wie dies beispielsweise bei einigen der untersuchten Pattern-Sammlungen der Fall ist (siehe Kapitel 3.3.2). Wenn Muster aus diesen in PaMGIS importiert werden, so können die betreffenden Informationen im Element <RLTN_Information> untergebracht werden. Präziseres Referenzieren von Patterns im Vergleich zu PLML 1.1, das auch automatisiert verarbeitet werden kann, kann mit dem Element „Relations“ erfolgen. Hierbei werden Versionen und Revisionen von Mustern einbezogen und es erfolgt zudem die Spezifikation des Typs der Beziehungen.

Alle Unterelemente des Elements <Relationship> sind zusammen mit ihren Beschreibungen in Abbildung 56 aufgelistet.

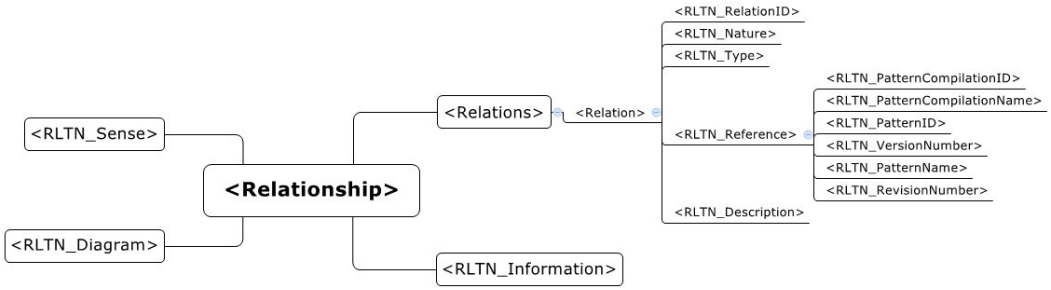


Abbildung 56 IM TOP-LEVEL-ELEMENT <RELATIONSHIP> ZUSAMMENGEFASSTE PPSL-BESCHREIBUNGSELEMENTE

Das Beschreibungselement <Deployment> ist eine Weiterentwicklung des PLML 1.1-Elements <Implementation>, das dort noch gänzlich unstrukturiert war. Es besteht aus den drei Unterelementen <Implementations>, <EmbeddingLinks> und <PaMGIS>. Das Element <Implementations> stellt die Kompatibilität zu anderen Pattern-Beschreibungssprachen her. Im Falle des Imports von Mustern in PaMGIS können die betreffenden Informationen in die hier vorgegebene Unterstruktur eingetragen werden. <EmbeddingLinks> kann, wie in [204] beschrieben, eine Liste von Rückverweisen auf OOA- und OOD-Modelle aufnehmen. Im Element <PaMGIS> sind schließlich die zum Muster gehörenden Aufgaben-, Konzept-, und Dialog-Modell-Fragmente, die bei der Anwendung des Patterns automatisch in die Modelle des modellgetriebenen Anteils von PAMGIS integriert werden können. Sie dienen hier quasi als Bausteine bei der Erstellung der Modelle und realisieren auf diese Art und Weise eine Wiederverwendung von Teilmodellen.

Die Beschreibungen aller Unterelemente des Elements <Deployment> sind in Abbildung 57 ersichtlich.

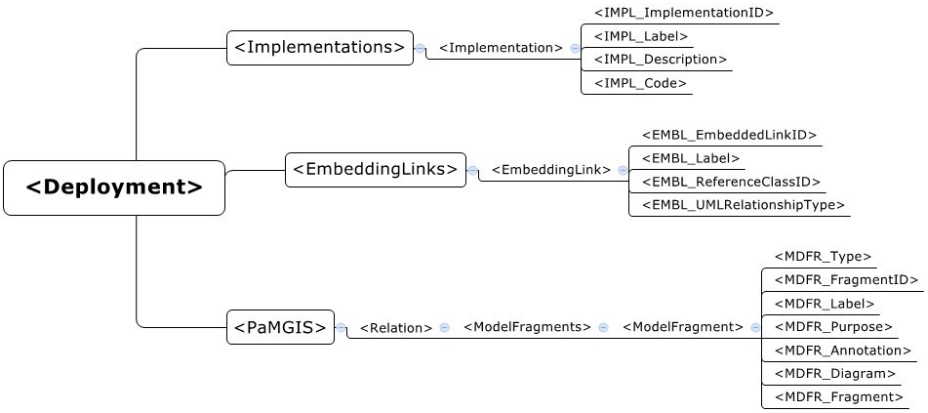


Abbildung 57 IM TOP-LEVEL-ELEMENT <DEPLOYMENT> ZUSAMMENGEFASSTE PPSL-BESCHREIBUNGSELEMENTE

Auflistungen aller Beschreibungselemente von PPSL mit Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befinden sich in Anhang G.1.

4.7.2.2. Beschreibung der Modell-Fragmente

In diesem Kapitel wird der Aufbau der in die Patterns eingebundenen Modell-Fragmente, die als Bausteine bei der Konstruktion der PaMGIS-Modelle dienen können, erklärt. Es handelt sich dabei um Fragmente für die Aufgaben-, Konzept- und Dialog-Modelle. In der PPSL-Struktur sind die Modell-Fragmente unter <Deployment> - <PaMGIS> - <ModelFragments> - <ModelFragment> abgelegt. Hier wird jeweils der Typ des Modell-Fragments, ein eindeutiger Bezeichner und eine Bezeichnung definiert. Darüber hinaus stehen zwei Elemente für die Beschreibung des Verwendungszwecks und sonstige Anmerkungen zur Verfügung. Die eigentlichen Inhalte der Fragmente werden mithilfe des Elements <MDFR_Fragment> spezifiziert.

4.7.2.2.1. Aufgaben-Modell-Fragment

Ein Pattern kann optional ein Aufgaben-Modell-Fragment enthalten, d.h. es ist entweder kein oder genau ein Fragment dieses Typs vorhanden. In der Regel verfügt es dann auch über genau ein Konzept-Modell-Fragment, das wechselseitige Beziehungen zum Aufgaben-Modell-Fragment aufweist. Auf der Grundlage der Lösungsbeschreibung des Musters spezifiziert es die hierbei durch den Benutzer durchzuführenden Aktivitäten.

Für die Beschreibung eines Aufgaben-Modell-Fragments stehen drei Unterelemente zur Verfügung. Mit dem Element <TMF_IncludesDummy> wird angezeigt, ob das Modell-Fragment variable Anteile in Form von sogenannten Dummy-Aufgaben enthält. Ist dies der Fall, so wird vor der Anwendung des Musters mithilfe des PaMGIS-Werkzeugs *Pattern / Model Selection & Assignment* ein Dialog geöffnet, der den Framework-Benutzer dabei unterstützt, die variablen Modellanteile zu konkretisieren. Die praktische Nutzung von Dummies ist im Rahmen der Fallstudie in Kapitel 5 beschrieben. Ferner existiert das Element <TMF_Content>, das im Wesentlichen die gleichen Beschreibungselemente wie das <Task>-Element im Aufgaben-Modell umfasst (siehe Abbildung 41 in Kapitel 4.6.2.1.1). Das Wurzel-Element wird jedoch im Fragment als <Subtask> spezifiziert, da es bei der Integration in ein Aufgaben-Modell als Unteraufgabe in den entsprechenden CTT-Baum eingetragen wird. Falls das Wurzel-Element des Fragments im Modell dann über ein linkes Nachbar-Element verfügt, kann dort die im Element <TMF_ProposedTempOp> vorgeschlagene zeitliche Beziehung in das Element <TemporalOperator> übernommen werden. Da es sich hierbei um einen Vorschlag handelt, wird hierfür im Werkzeug *Pattern / Model Selection & Assignment* (siehe Kapitel 4.6.1) bei der Anwendung des Patterns ein entsprechender Dialog geöffnet, damit der Benutzer wählen kann, ob er den Vorschlag annehmen möchte oder nicht.

Darüber hinaus enthält das Fragment im Element <TaskID> einen Bezeichner der Aufgabe, der nicht unbedingt dem späteren Bezeichner im Aufgaben-Modell entsprechen muss. Deshalb ist dieser bei der Integration des Fragments anzupassen. Ebenso könnte der Benutzer auch einen anderen Namen für die Aufgabe vergeben wollen. Beide Informationen müssen daher im oben genannten User-Dialog bearbeitet werden können.

Im Element <TaskOrigin> und seinen Unterelementen sind die entsprechenden Informationen desjenigen Patterns, welches das Fragment enthält, bereits eingetragen. Dieser Teil kann also unverändert in das Aufgaben-Modell übernommen werden.

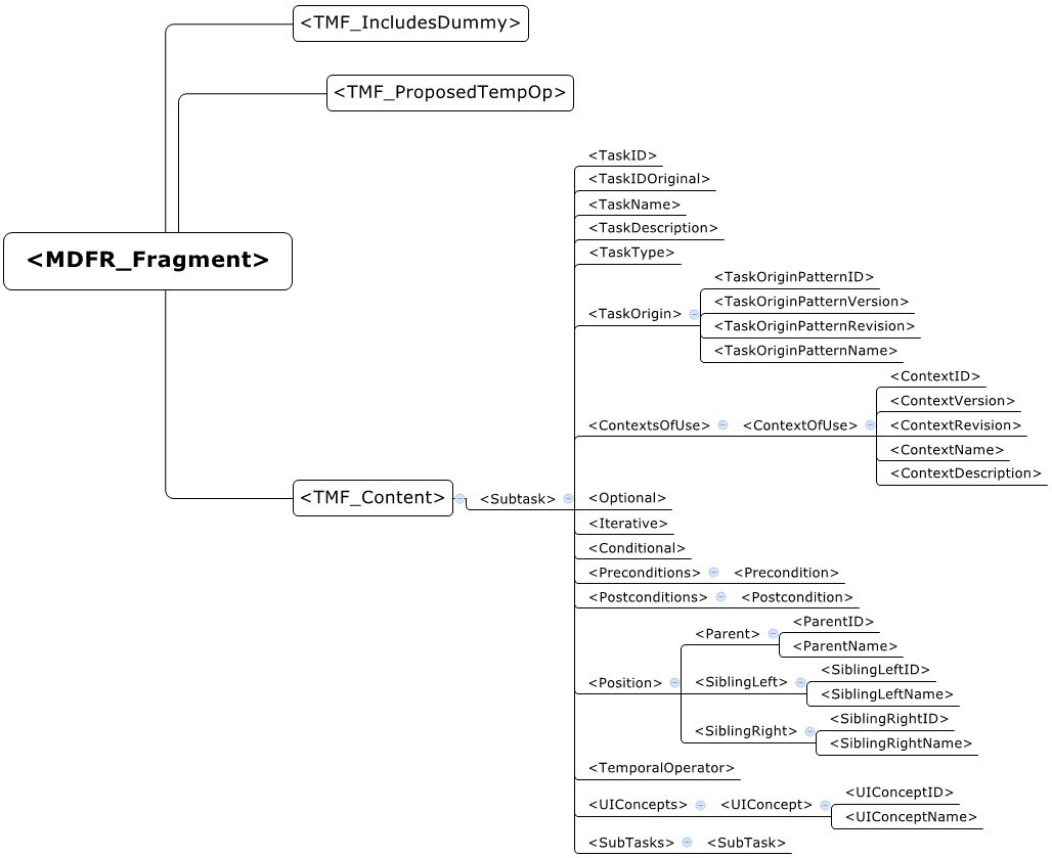


Abbildung 58 AUFBAU EINES AUFGABEN-MODELL-FRAGMENTS <MDFR_FRAGMENT>

Das Element <ContextsOfUse> enthält eine Liste von Kontexten, in denen die Aufgabe auftauchen soll. Hierbei gibt es zwei Möglichkeiten, die Kontexte zu beschreiben. Einerseits kann dies durch entsprechende Referenzen zu vorhandenen, archetypischen Kontext-Modellen geschehen. Dafür dienen Beschreibungselemente <ContextID>, <ContextVersion> und <ContextRevision>. Ist dies nicht der Fall, so kann die Beschreibung andererseits auch in textueller Form im Element <ContextDescription> vorliegen. Dann muss der User entscheiden, ob er diese unverändert übernehmen möchte.

Mithilfe der Angaben im <Position>-Element wird die Verankerung des Wurzel-Elements des Fragments im Aufgaben-Modell bestimmt. Vor der Integration des Fragments muss der Benutzer die Stelle im Modell festgelegt haben, an der das Teilmodell aus dem Pattern eingefügt werden soll. Ist dies geschehen, so können die benötigten Informationen hinsichtlich des Eltern- und der Nachbar-Elemente automatisch eingetragen werden.

Im Beschreibungselement <UIConcept> werden, wie beim Aufgaben-Modell auch, die zur Aufgabe gehörenden Konzepte referenziert. Im Fragment beziehen sich die Referenzen aber auf das korrespondierende Konzept-Modell-Fragment. Bei der Integration in das Aufgaben-Modell sind auch diese Verweise entsprechend anzupassen (siehe Kapitel 4.7.2.2.2).

Eine Übersicht über alle Elemente des Aufgaben-Modell-Fragments wird in Abbildung 58 gegeben. Eine vollständige Liste der Elemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang G.2.1.

4.7.2.2.2. Konzept-Modell-Fragment

Wie bereits erwähnt, treten Konzept- und Aufgaben-Modell-Fragmente in der Regel paarweise in einem Pattern auf. Dabei beinhaltet das Konzept-Modell-Fragment diejenigen UI-Konzepte, die für die Aufgaben im Aufgaben-Modell-Fragment benötigt werden. Wie auch schon innerhalb des Domänen-Modells, stehen die beiden Fragmente in enger Beziehung zueinander, d.h. es bestehen beiderseitige Referenzen von Aufgaben und zugehörenden Konzepten.

Die Beschreibung eines Konzept-Modells besteht aus den beiden Elementen <CMF_IncludesDummy> und <CMF_Content>. Ersteres dient als Kennzeichnung, ob das Konzept-Modell-Fragment einen oder mehrere Dummy-Konzepte zur interaktiven Gestaltung variabler Modellanteile einhält. Innerhalb des letzteren Elements können ein oder mehrere Konzepte spezifiziert werden. Hierzu stehen die gleichen Unterelemente zur Verfügung wie beim <Concept>-Element im Konzept-Modell (siehe Abbildung 43 in Kapitel 4.6.2.1.2).

Da die Auflistung der Konzepte im Konzept-Modell unsortiert erfolgt, können deshalb beim Einfügen des Fragments die dort definierten Konzepte grundsätzlich einfach an das Ende der Liste im Konzept-Modell angehängt werden. Zuvor müssen aber, analog zu den Aufgaben-Modell-Fragmenten, jeweils der eindeutige Bezeichner und, sofern der Benutzer dies wünscht, auch der Name und das Label angepasst werden. Hierfür muss wiederum im Tool *Pattern Selection and Assignment* ein interaktiver Dialog bereitgestellt werden.

Im Element <TaskOrigin> und seinen Unterelementen sind die entsprechenden Informationen desjenigen Patterns, welches das Fragment enthält, wiederum bereits eingetragen. Es kann also unverändert in das Konzept-Modell übernommen werden.

Im Beschreibungselement <CCPT_TaskLinks> werden die Aufgaben, die das betreffende Konzept benötigen, referenziert. Im Konzept-Modell-Fragment beziehen sich die Referenzen auf das korrespondierende Aufgaben-Modell-Fragment. Bei der Integration in das Konzept-Modell sind also diese Verweise entsprechend anzupassen.

Grundsätzlich sind bei der Integration eines Konzepts aus einem Fragment in ein Konzept-Modell zwei verschiedene Fälle zu unterscheiden. Falls ein entsprechendes Konzept noch nicht im Konzept-Modell vorhanden ist, kann dieses wie oben beschrieben eingefügt werden. Gibt es aber bereits ein Konzept, das der Benutzer für diesen Zweck verwenden möchte, so muss dieses entsprechend modifiziert werden, insbesondere sind die gegenseitigen Referenzen zwischen Konzept und Aufgabe entsprechend anzupassen. Dies erfolgt wiederum mithilfe eines interaktiven Dialogs.

Eine Übersicht über alle Elemente des Konzept-Modell-Fragments ist in Abbildung 59 ersichtlich. Eine vollständige Liste der Elemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang G.2.2.

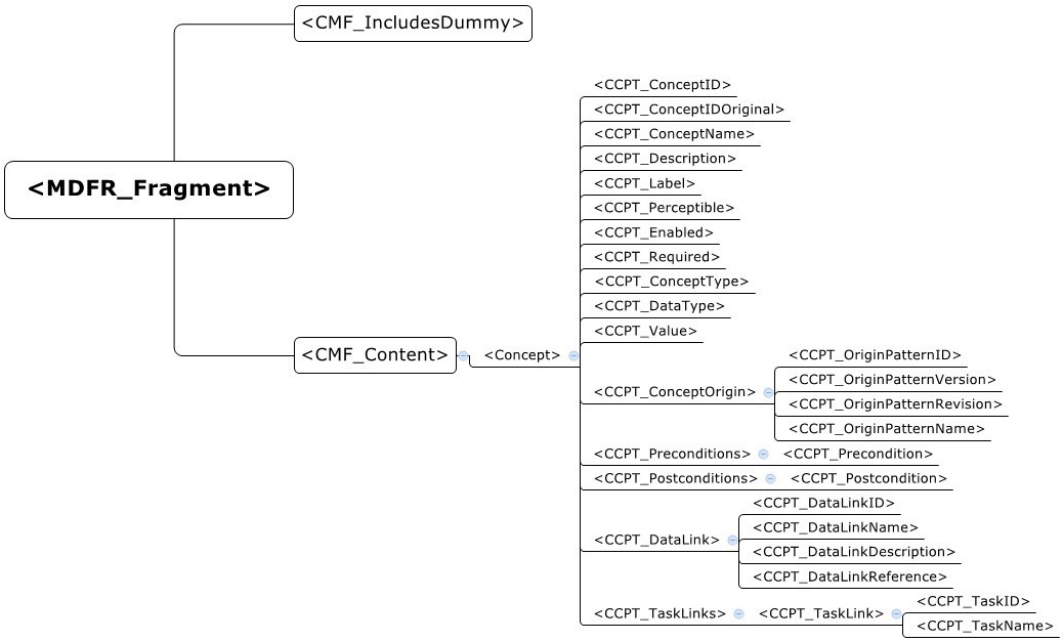


Abbildung 59 AUFBAU EINES KONZEPT-MODELL-FRAGMENTS <MDFR_FRAGMENT>

4.7.2.2.3. Dialog-Modell-Fragment

Während Aufgaben- und Konzept-Modell-Fragmente in der Regel höchstens einmal in einem Muster auftreten, können durchaus mehrere Dialog-Modell-Fragmente in einer Pattern-Spezifikation enthalten sein. Dies ist dann der Fall, wenn Dialog-Strukturen für verschiedene Benutzungskontexte hinterlegt wurden, also beispielsweise für unterschiedliche Endgeräte wie Smartphone und Desktop-PC.

Zur Beschreibung eines Dialog-Modell-Fragments steht das Element <DMF_Content> zur Verfügung. Innerhalb dieses Elements können ein oder mehrere Dialoge spezifiziert werden. Es besteht aus den gleichen Unterelementen wie das <Dialog>-Element des Dialog-Modells (siehe Abbildung 44 in Kapitel 4.6.2.2). Optional können ein oder mehrere Kontexte-Modelle referenziert werden, für die das Modell-Fragment anwendbar sein soll. Dies erfolgt mithilfe des Elements <DMF_ContextModelReferences>.

Auch bei diesem Modell- Fragment-Typ müssen wieder der eindeutige Bezeichner des Dialogs und, sofern gewünscht, auch der betreffende Name und das Label angepasst werden. Dies erfolgt wiederum mittels eines interaktiven Dialogs beim Einfügen des Fragments in das Dialog-Modell.

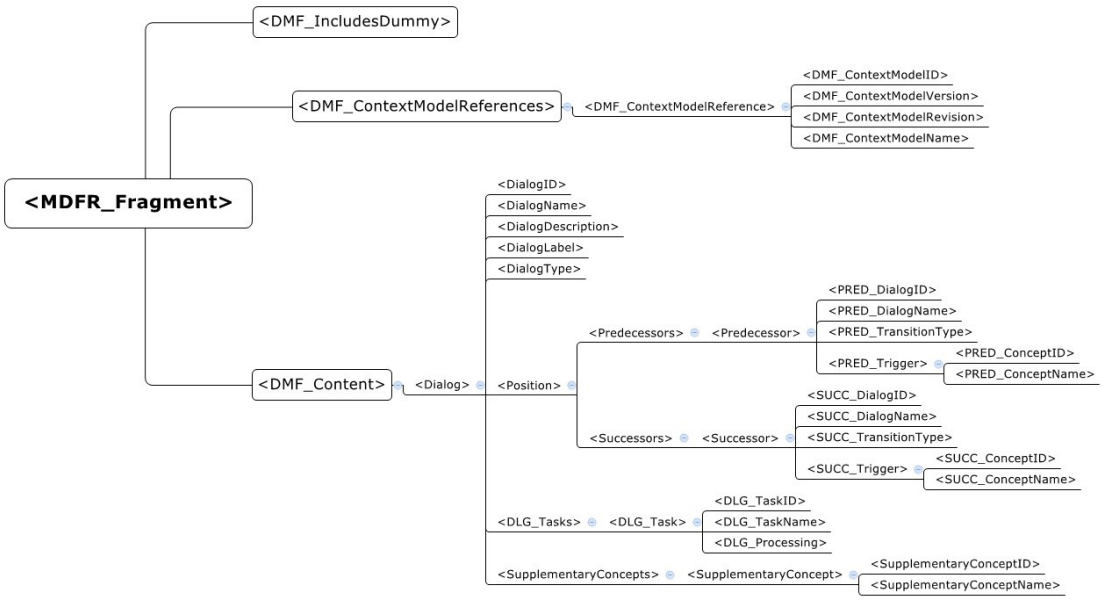


Abbildung 60 AUFBAU EINES DIALOG-MODELL-FRAGMENTS <MDFR_FRAGMENT>

Die Dialoge sind, analog zum Dialog-Modell, mithilfe des <Position>-Elements vorwärts und rückwärts verkettet. Je nach der vorliegenden Struktur im Dialog-Modell muss diese Verkettung jeweils für den Start- und den End-Dialog des Fragments zur Einbindung in den globalen Dialogfluss angepasst werden. Da sich die eindeutigen Bezeichner und gegebenenfalls auch die Namen bei der Integration ins Gesamt-Modell geändert haben können, müssen auch bei den übrigen Dialogen die entsprechenden Informationen im <Position>-Element adaptiert werden.

Im Beschreibungselement <DLG_Tasks> werden die Aufgaben, die dem betreffenden Dialog zugeordnet werden sollen, referenziert. Im Dialog-Modell-Fragment beziehen sich diese Referenzen auf das korrespondierende Aufgaben-Modell-Fragment. Bei der Integration ins das Dialog-Modell müssen auch diese Verweise entsprechend angepasst werden.

Die zusätzlichen Konzepte zur Navigation zwischen den Dialogen, die keine korrespondierenden Aufgaben im Task-Modell besitzen, werden mithilfe des Beschreibungselements <SupplementaryConcepts> einem Dialog zugeordnet.

Eine Übersicht über alle Elemente des Dialog-Modell-Fragments wird in Abbildung 60 gezeigt. Eine vollständige Liste der Elemente einschließlich der Erklärungen und Angaben zu den jeweiligen Datentypen und Kardinalitäten befindet sich in Anhang G.2.3.

5. Fallstudie

Anhand der in diesem Kapitel ausgeführten Fallstudie wird die praktische Arbeitsweise mit dem PaMGIS-Framework aufgezeigt und erklärt. Sie dient zudem als Nachweis dafür, dass sich die Erstellung der Modelle für den modellgetriebenen Anteil des Frameworks durch die Verwendung der um die Modell-Fragmente erweiterten Muster vereinfachen lässt.

Zunächst werden das Thema und der Umfang der Fallstudie definiert. Danach erfolgt die Erstellung und Spezifikation einer geeigneten Pattern-Sprache. Anschließend wird beschrieben, wie die entsprechenden Benutzungskontext-Modelle erstellt werden. Im nächsten Schritt erfolgt die Konstruktion des Domänen-Modells mithilfe der Muster der Pattern-Sprache. Dann werden, ebenfalls mit Unterstützung durch die in den Mustern enthaltenen Informationen, die Dialog-Modelle aufgebaut. Letztlich wird noch skizziert, wie aus den bis dahin konstruierten Modellen die schrittweise Ableitung der abstrakten (AUI), konkreten (CUI) und schließlich der finalen Benutzerschnittstelle (FUI) vonstattengeht.

5.1. Thema und Umfang der Fallstudie

Die Fallstudie beschäftigt sich mit der Modellierung und Generierung von Benutzeroberflächen von Systemen zum Verkauf von Tickets zur Personenbeförderung. Es wird dabei bewusst darauf verzichtet, die Art der Verkehrsmittel genauer festzulegen. Dies geschieht erst zu einem späteren Zeitpunkt, nämlich beim Design der Domänen- und Dialog-Modelle. Insbesondere bedeutet das, dass die zum Einsatz kommenden Patterns so gestaltet sein müssen, dass sie die Modellierung von Benutzerschnittstellen jedweder Systeme für den Reiseticketverkauf gleichermaßen unterstützen, also beispielsweise sowohl für den Nah- und Fernverkehr als auch für Flugreisen oder Bahn-, Tram- und Busfahrten.

Es werden zwei verschiedene Benutzungskontexte berücksichtigt, die sich hauptsächlich hinsichtlich der zum Einsatz kommenden Benutzerendgeräte unterscheiden. Die Modellierung erfolgt beispielhaft für den Bahnverkehr. Die Realisierung der Benutzeroberflächen basiert auf der Programmiersprache *Python* in Kombination mit *Tkinter*, einem für Python angepassten Toolkit für graphische User-Interfaces.

5.2. Definition der Patternsprache

Grundsätzlich könnten mit dem PaMGIS-Framework auch gänzlich ohne Verwendung von Mustern Modelle erstellt und Benutzeroberflächen generiert werden. Damit man aber die Vorteile des kombinierten, modellgetriebenen und musterbasierten Entwicklungsansatzes nutzen kann, benötigt man eine für die Anwendungsdomäne passende Pattern-Sprache. Diese wird im vorliegenden Kapitel beschrieben.

Zunächst wurden mehrere existierende Verkaufssysteme für Reisetickets eingehend untersucht. Darunter befanden sich unter anderem das Mobilitätsportal der Deutschen Bahn AG[191], die Smartphone-App der Deutschen Bahn AG[192], das internationale Internet-Portal der Deutschen Bahn AG[193], ein Fahrkartenautomat des Augsburger Verkehrsverbunds[194] und das Internetportal der Deutschen Lufthansa AG[195]. Insbesondere von diesen Systemen wurden bei der späteren Beschreibung der Patterns Screenshots und Fotos verwendet.

Bei dieser Analyse wurden insgesamt dreizehn Muster identifiziert, die in die zu erstellende Pattern-Sprache integriert wurden.

Tabelle 25 MUSTER DER PATTERN-SPRACHE FÜR DEN TICKETVERKAUF ZUR PERSONENBEFÖRDERUNG

Name	Beschreibung
Reiseticket kaufen	Durchführung des Kaufs von Reisetickets. Dabei sind stets mindestens folgende Schritte durchzuführen: (1) Angabe der Reisedaten, (2) Auswahl der gewünschten Reisverbindung, (3) Spezifizieren der Ausführungsart, d. h. der angebotenen Zahlungs- und Versandmodalitäten, (4) Abschließen des Ticketkaufs.
Reisedaten spezifizieren	Es sind diejenigen Reisedaten festzulegen, die für die Suche nach den gewünschten Reiseverbindungen herangezogen werden sollen.
Reisestrecke spezifizieren	Es sind diejenigen Daten festzulegen, die die gewünschte Reisestrecke beschreiben.
Reisezeit spezifizieren	Es sind diejenigen Daten festzulegen, die die gewünschte Reisezeit beschreiben.
Reisende spezifizieren	Es sind die tariflich relevanten Eigenschaften der zu befördernden Personen und gegebenenfalls mitzunehmenden Dinge festzulegen.
Reiseoptionen spezifizieren	Es sind die buchbaren Reiseoptionen, wie beispielsweise die Reiseklasse, festzulegen.
Reiseverbindung wählen	Die gewünschte Reiseverbindung ist festzulegen.
Ausführungsart spezifizieren	Es sind diejenigen Daten festzulegen, die für die Ausführung des Ticketkaufs erforderlich sind. Dazu zählen: (1) Zahlungsart, (2) Versandart für die Tickets
Zahlungsart spezifizieren	Es ist die Zahlungsart für den Ticketkauf festzulegen.
Versandart spezifizieren	Es ist die Versandart für die Tickets festzulegen.
Ticketverkauf abschließen	Es sind diejenigen Aktionen durchzuführen, die für den Abschluss des Ticketkaufs erforderlich sind.
Eingaben überprüfen	Der Käufer soll vor der verbindlichen Bestellung der Tickets die Möglichkeit erhalten, die relevanten Angaben und ausgewählten Reisen noch einmal zu überprüfen.
Wizard	Benutzern wird die Bewältigung komplexer Aufgaben dadurch erleitert, indem ihnen die vorhandenen Unteraufgaben Schritt für Schritt präsentiert und zur Abarbeitung angeboten werden.

[191] https://www.bahn.de
[192] Smartphone-App *DBTickets* für iOS-Geräte
[193] https://www.deutsche-bahn.com/de
[194] Fahrscheinautomat des Augsburger Verkehrsverbunds (AVV) in Klosterlechfeld
[195] https://www.lufthansa.com/de

Neben diesen Mustern wurde zu Demonstrationszwecken noch das „Wizard“-Pattern aus der Sammlung *Patterns in Interaction Design* von Martijn van Welie [183] hinzugefügt. Tabelle 25 gibt einen Überblick über die verwendeten Muster.

An dieser Stelle wird nun beispielhaft näher auf das Muster „Reisezeit spezifizieren“ eingegangen. Das Pattern dient dazu, die Reisezeitpunkte festzulegen. In Tabelle 26 wird ein Auszug aus der Beschreibung dieses Musters dargestellt.

Tabelle 26 AUSZUG AUS DER BESCHREIBUNG DES MUSTERS „REISEZEIT SPEZIFIZIEREN“

Beschreibung	Inhalt
Problem	Es sind diejenigen Daten zu spezifizieren, die die gewünschte Reisezeit festlegen.
Kontext	Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder Flugzeug.
Lösung	Folgende Angaben zur Reisezeit werden benötigt: 1. Angabe des Hinreisezeitpunkts 1.1. Datum der Hinreise 1.2. Uhrzeit der Hinreise 1.3. Angabe, ob der angegebene Hinreise-Zeitpunkt als Abfahrts zeit oder Ankunftszeit zu interpretieren ist 2. Optionale Angabe des Rückreisezeitpunkts 2.1. Datum der Rückreise 2.2. Uhrzeit der Rückreise 2.3. Angabe, ob der angegebene Rückreise-Zeitpunkt als Abfahrts zeit oder Ankunftszeit zu interpretieren ist

Die Verwendung des Patterns im Mobilitätsportal der Deutschen Bahn AG wird als Anwendungsbeispiel in Abbildung 61 gezeigt.

Abbildung 61 VERWENDUNG DES MUSTERS „REISEZEIT SPEZIFIZIEREN“ IM MOBILITÄTSPORTAL DER DEUTSCHEN BAHN AG

In der PPSL-Repräsentation des Musters befinden sich die Informationen aus Tabelle 26 in den Beschreibungselementen <Body> <Theory> <Problem>, <Body> <Theory> <Context> bzw. <Body> <Theory> <Solution>, die Grafik aus Abbildung 61 ist im Beschreibungselement <Body> <Practice> <KnownUses> abgelegt. Die vollständige PPSL-Spezifikation kann in Anhang J.4 eingesehen werden.

Das dem Pattern inhärente Aufgaben-Modell wurde auf die bereits in Kapitel 4.4.4.2 beschriebene Art und Weise erstellt und auf die bei der Generierung der Benutzeroberfläche tatsächlich benötigten Anteile reduziert. Wie in Abbildung 62 gezeigt, repräsentiert das Wurzel-Element des zugehörigen CTT-Baums die Aufgabe „Reisezeit spezifizieren“, die aus zwei Teilaufgaben besteht. Dies ist einerseits die obligatorische Aufgabe „Hinreisezeit spezifizieren“, die immer ausgeführt werden muss und andererseits die bedingt auszufüh-

rende Aufgabe „Rückreise-Zeitpunkt spezifizieren“, die nur dann abgearbeitet wird, wenn die Rückreise tatsächlich gewünscht wird.

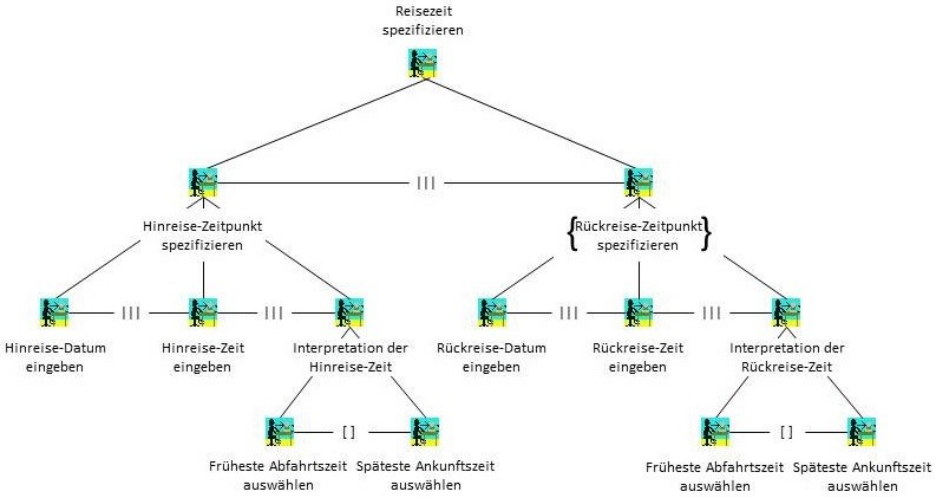


Abbildung 62 INHÄRENTES AUFGABEN-MODELL DES MUSTERS „REISEZEIT SPEZIFIZIEREN“

Die beiden genannten Teilaufgaben bestehen ihrerseits wieder jeweils aus Teilaufgaben zur Festlegung des Datums und der Uhrzeit sowie zur Angabe, ob dieser Zeitpunkt die Abreise- oder die Ankunftszeit darstellen soll. Dieses Modell wird in Form eines Aufgaben-Modell-Fragments in der PPSL-Spezifikation des Patterns hinterlegt. Dazu wird das Beschreibungselement <Deployment> <PaMGIS> <ModelFragments> verwendet.

In Tabelle 27 befindet sich die Liste der vom Pattern benötigten Konzepte. Für die Eingabe von Datum und Uhrzeit kommt jeweils ein Konzept des Typ *DataEdit* zum Einsatz. Für die Interpretationen der Zeitangaben werden *SingleChoice*-Konzepte verwendet. Die jeweils auswählbaren Optionen werden durch Konzepte des Typs *ChoiceItem* repräsentiert.

Tabelle 27 ÜBERSICHT ÜBER DIE BENÖTIGTEN KONZEPTE DES MUSTERS „REISEZEIT SPEZIFIZIEREN“

Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
Hinreisedatum	DataEdit	Hinreise-Datum eingeben
Hinreisezeit	DataEdit	Hinreise-Zeit eingeben
Interpretation Hinreisezeit	SingleChoice	Interpretation der Hinreise-Zeit
Hinreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Hinreise)
Hinreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Hinreise)
Rückreisedatum	DataEdit	Rückreise-Datum eingeben
Rückreisezeit	DataEdit	Rückreise-Zeit eingeben
Interpretation Rückreisezeit	SingleChoice	Interpretation der Rückreise-Zeit
Rückreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Rückreise)
Rückreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Rückreise)

Die Liste der Konzepte wird in Form eines Konzept-Modell-Fragments, ebenfalls mittels des Beschreibungselements <Deployment> <PaMGIS> <ModelFragments>, in der PPSL-Spezifikation des Musters gespeichert. Dabei werden für jedes Konzept unter Zuhilfenahme des Be-

schreibungselements <CCPT_TaskLinks> diejenigen Aufgaben des Task-Modell-Fragments referenziert, die das Konzept verwenden. Andererseits verfügen auch die Aufgaben jeweils über Links auf die ihnen zugeordneten Konzepte.

Auf Basis des Task-Modell-Fragments werden dann noch ein oder mehrere Dialog-Modell-Fragmente erstellt. Die Vorgehensweise folgt dabei demselben Prinzip, wie es bereits in Kapitel 4.4.4.4 für das „Poll"-Pattern erläutert wurden. Im Rahmen der Fallstudie wurde entschieden, für das Muster „Reisezeit spezifizieren" nur ein einziges DMF zu definieren. Dieses kann dann für alle in Frage kommenden Benutzungskontexte eingesetzt werden. Der entsprechende Dialog-Graph wird in Abbildung 63 illustriert. Er umfasst die beiden Aufgaben „Hinreise-Zeitpunkt spezifizieren" und „Rückreise-Zeitpunkt spezifizieren" mit allen ihren Unteraufgaben.

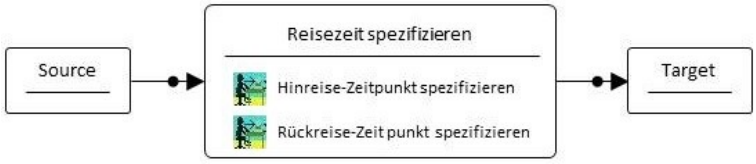


Abbildung 63 DIALOG-MODEL-FRAGMENT DES MUSTERS „REISEZEIT SPEZIFIZIEREN"

Das DMF wird in gleicher Weise, wie auch die anderen Modell-Fragmente, in der Pattern-Spezifikation hinterlegt.

Alle anderen Muster der Pattern-Sprache (siehe Tabelle 25) wurden in analoger Weise definiert und mithilfe von PPSL formal beschrieben. Die Beziehungen, in denen die Patterns jeweils zueinander stehen, werden in Abbildung 64 visualisiert.

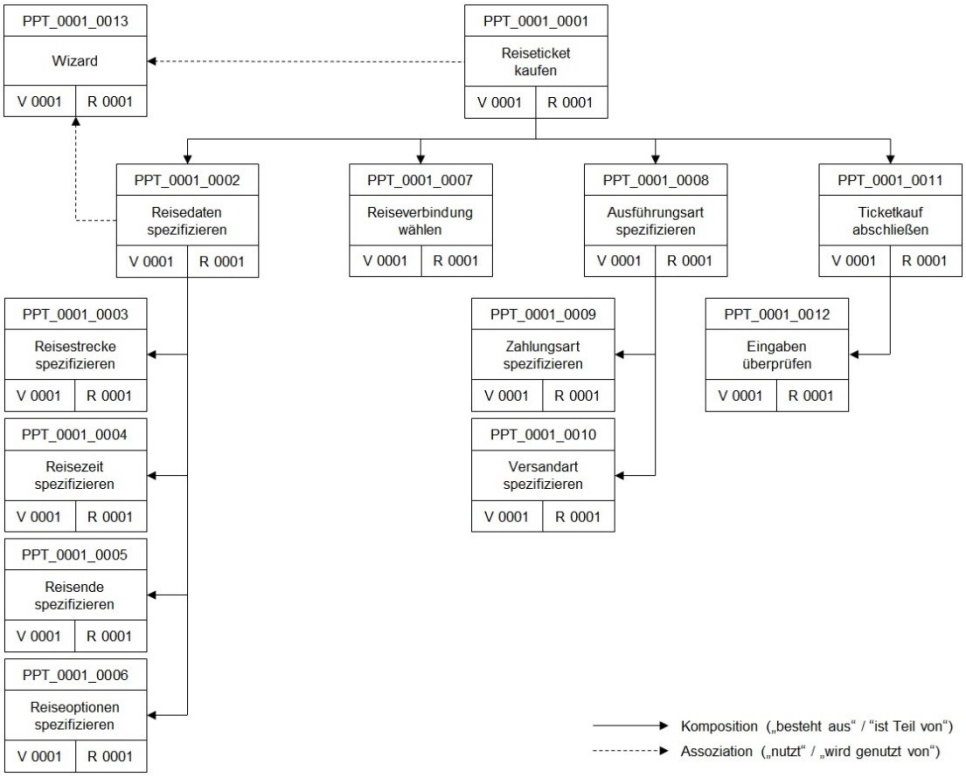


Abbildung 64 BEZIEHUNGEN ZWISCHEN DEN MUSTERN DER PATTERN-SPRACHE

Die Pfeile mit durchgezogenen Linien kennzeichnen hierbei Kompositionen, d.h. sie stehen für die Beziehung „besteht aus“ bzw. „ist Teil von“. Pfeile mit gestrichelten Pfeilen repräsentieren die vergleichsweise losere Beziehung „nutzt“ bzw. „wird genutzt von“. Diese wurde im Rahmen der Fallstudie hauptsächlich bei der Erstellung der Dialog-Modelle herangezogen.

Die vollständige PPSL-Spezifikation der Pattern-Sprache für die Domäne des Verkaufs von Tickets für die Personenbeförderung befindet sich in Anhang J.

5.3. Erstellung der Kontext-Modelle

Wie bereits vorher erwähnt, werden in der Fallstudie zwei Benutzungskontexte betrachtet. Grundsätzlich können sich die Kontexte bei der Verwendung von PaMGIS beliebig hinsichtlich der Eigenschaften der Benutzer, Endgeräte, UI-Toolkit und Umgebung unterscheiden. Aus Gründen der Übersichtlichkeit und zur besseren Nachvollziehbarkeit werden hier aber nur die Spezifikationen der zum Einsatz kommenden Benutzerendgeräte variiert. Im Wesentlichen unterscheiden sich diese bezüglich ihrer visuellen Ausgabemöglichkeiten. Der erste Kontext umfasst das Benutzer-Modell eines „Standard-Benutzers“, das Geräte-Modell eines „Standard-Desktop-PC“, das UI-Toolkit-Modell für „Python/Tkinter“ und das Umgebungs-Modell für eine „Standard-Büroumgebung“. Was darunter genau zu verstehen ist, wird im Folgenden beschrieben. Eine grafische Übersicht über das betreffende Kontextmodell wird in Abbildung 65 gezeigt.

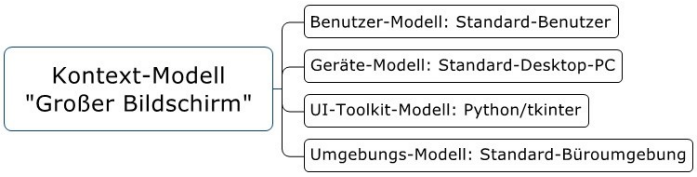


Abbildung 65 ÜBERSICHT ÜBER DAS KONTEXT-MODELL „GROßER BILDSCHIRM“

Das Hauptmerkmal ist dabei der Einsatz eines bei solchen Geräten üblichen großen Monitors, weshalb der Kontext auch die Bezeichnung „Großer Bildschirm“ trägt. Der zweite Kontext beinhaltet zwar dieselben Benutzer-, UI-Toolkit- und Umgebungs-Modelle, berücksichtigt aber das Geräte-Modell für ein typisches Smartphone. Das Augenmerk liegt hierbei auf dem im Vergleich deutlich kleineren Display, weshalb der Kontext „Kleiner Bildschirm“ genannt wird. Der Aufbau dieses Benutzungskontexts ist in Abbildung 66 illustriert.

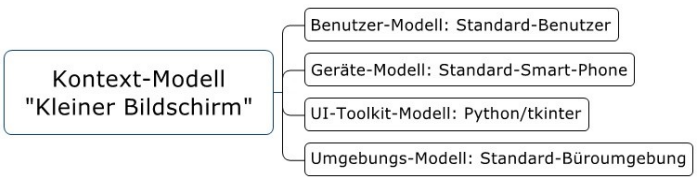


Abbildung 66 ÜBERSICHT ÜBER DAS KONTEXT-MODELL „KLEINER BILDSCHIRM“

Die vollständigen XML-Repräsentationen der beiden Kontext-Modelle können in den Anhängen K.1 und K.2 eingesehen werden.

Das Benutzer-Modell beschreibt einen User ohne signifikante körperliche und geistige Einschränkungen sowie mit fortgeschrittenen Kenntnissen sowohl hinsichtlich der Anwen-

dungsdomäne als auch des Umgangs mit diesbezüglichen Systemen. Er erfährt wenig Ablenkung und kann sich deshalb gut auf die Bedienung der Benutzeroberfläche konzentrieren. Es handelt sich um eine voll geschäftsfähige Person, die deswegen auch den Ticketkauf rechtskräftig abschließen darf. Die Eigenschaften des Benutzers sind in Abbildung 67 dargestellt.

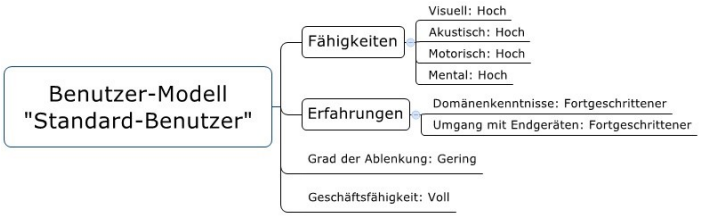


Abbildung 67 DETAILS DES BENUTZER-MODELLS „STANDARD-BENUTZER“

Der im Kontext „großer Bildschirm“ verwendete Standard-Desktop-PC besitzt das Betriebssystem *Windows 7* und für Eingaben eine Tastatur und ein Zeigegerät, also etwa eine Maus oder einen Trackball. Die visuellen Ausgaben erfolgen über einen 24-Zoll-Bildschirm mit der Auflösung *Full HD*, die akustischen über einen Lautsprecher. Die Merkmale dieses Endgeräts sind in Abbildung 68 zusammengefasst.

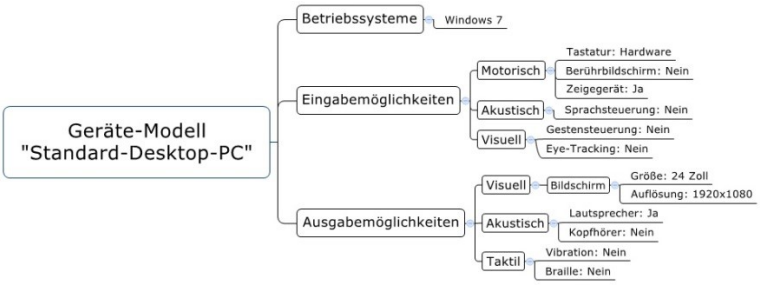


Abbildung 68 DETAILS DES GERÄTE-MODELLS „STANDARD-DESKTOP-PC“

Wie aus Abbildung 69 entnommen werden kann, läuft auch auf dem Standard-Smart-Phone das Betriebssystem Windows. Es ist mit einer, für diese Gerätegattung typischen, in Software realisierten Tastatur und einem Touch-Screen der Größe von fünf Zoll und einer Auflösung von 540 mal 690 Pixel ausgestattet. Es verfügt über einen Lautsprecher und der Möglichkeit, Vibrationsalarme zu erzeugen.

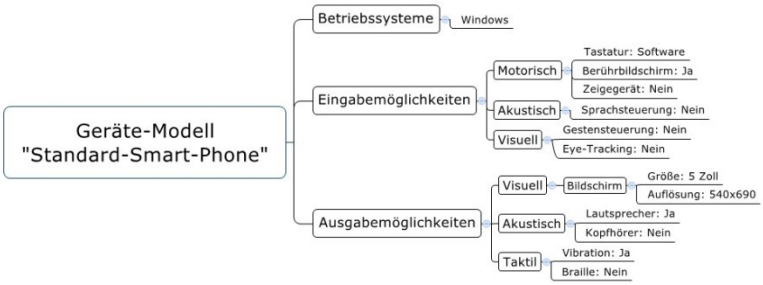


Abbildung 69 DETAILS DES GERÄTE-MODELLS „STANDARD-SMART-PHONE“

Das verwendete UI-Toolkit-Modell bezieht sich auf *Tkinter*, einer Schnittstelle der Programmierprache *Python*[196] zu *Tk*[197]. *Tk* wiederum ist ein freies, plattformübergreifendes Toolkit zur Erstellung von grafischen Benutzeroberflächen. *Tkinter* ist für Windows , Linux und einige weitere Betriebssysteme verfügbar. Es stellt 19 verschiedene Typen von UI-Elementen, wie beispielsweise etwa „Button“, „Checkbutton“, „Listbox“ oder „Radiobutton“ zur Verfügung. Eine Übersicht der unterstützten Betriebssysteme und der angebotenen UI-Elemente wird in Abbildung 70 gegeben.

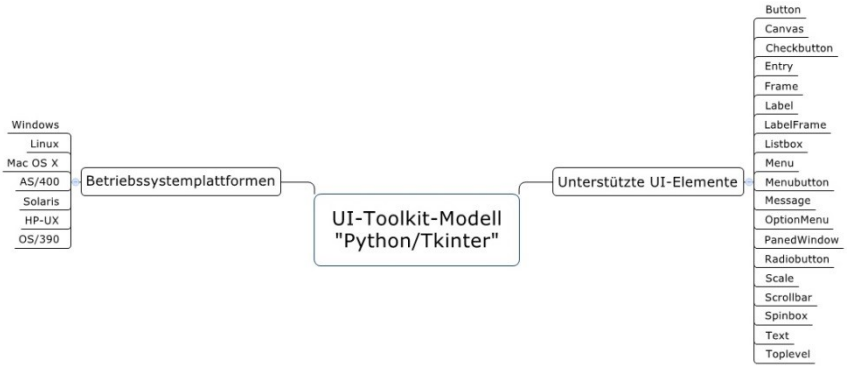


Abbildung 70 DETAILS DES UI-TOOLKIT-MODELLS „PYTHON/TKINTER“

Das in Abbildung 71 dargestellte Umgebungs-Modell definiert abschließend noch eine Standard-Büroumgebung, die sich durch für die Benutzer angenehme Arbeitsbedingungen, wie geringe negative Lichteinflüsse, geringe Umgebungsgeräusche und geringe Staubbelastung, auszeichnet.

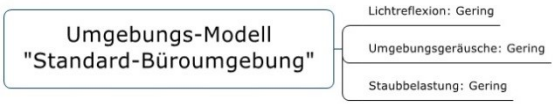


Abbildung 71 DETAILS DES UMGEBUNGS-MODELLS „STANDARD-BÜROUMGEBUNG“

Für alle in diesem Kapitel besprochenen Modelle können die entsprechenden vollständigen XML-Repräsentation in Anhang K eingesehen werden.

5.4. Erstellung des Domänen-Modells

Wie bereits erwähnt, soll die Erstellung des Domänen-Modells beispielhaft für den Kauf von Bahntickets erfolgen. Es sind also ein Aufgaben- und ein Konzept-Modell zu erstellen, wobei die jeweiligen Modell-Fragmente aus der in Kapitel 5.2 beschriebenen Pattern-Sprache als Bausteinmodule dienen. Da die entsprechenden Muster nicht ausdrücklich für Bahntickets, sondern generell für den Reiseticketverkauf entworfen wurden, müssen die Inhalte der Modell-Fragmente hierfür gegebenenfalls noch angepasst werden. Im Wesentlichen wird dies dadurch erreicht, dass vorhandene Dummy-Aufgaben und Dummy-Konzepte bei der

[196] Details siehe beispielsweise https://www.python.org/
[197] Der Name *Tk* steht abkürzend für das Wort *Toolkit*

Anwendung der Muster interaktiv durch den PaMGIS-Benutzer für den gewünschten Zweck adaptiert werden.

Als Ausgangspunkt für die Konstruktion des Domänen-Modells dient das in Abbildung 72 gezeigte Aufgaben-Modell, das unter Zuhilfenahme des Modell-Editors erstellt wird.

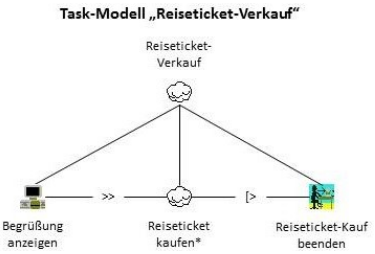


Abbildung 72 INITIALE VERSION DES AUFGABEN-MODELLS ZUM REISETICKETKAUF

Das Wurzel-Element ist eine abstrakte Aufgabe namens „Reiseticket-Verkauf“, die über drei Unteraufgaben verfügt. Diese sind eine Anwendungs-Aufgabe zur Anzeige eines Begrüßungsbildschirms, eine beliebig oft wiederholbare abstrakte Aufgabe zur Abwicklung des Ticketkaufs sowie eine Interaktions-Aufgabe zur Beendigung des Kaufvorgangs beziehungsweise des interaktiven Anwendungsprogramms.

Hierbei wird für die erste Unteraufgabe ein Konzept zur Anzeige eines Begrüßungstextes bzw. einer entsprechenden Grafik und für die letzte Unteraufgabe ein Konzept zur interaktiven Beendigung des Reiseticketkaufs benötigt. Diese sind in Tabelle 28 aufgelistet. Aus Gründen der Übersichtlichkeit und Nachvollziehbarkeit werden hier nur die grundlegenden Beschreibungselemente der Konzepte aufgeführt. Zudem sind die Konzepte im Folgenden durchnummeriert.

Tabelle 28 ÜBERSICHT ÜBER DIE VERWENDETEN KONZEPTE (AUSGANGSSITUATION)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
1	Begrüßung	DataOutput	Begrüßung anzeigen
2	Ticketkauf beenden	Activator	Reiseticket-Kauf beenden

Im nächsten Schritt wird das obige Domänen-Modell erweitert. Dazu wird die abstrakte Aufgabe „Reiseticket kaufen“ weiter detailliert, d.h. es werden entsprechende Unteraufgaben definiert. Die Modellierung kann entweder schrittweise manuell oder durch die Anwendung eines passenden Patterns erfolgen. In der in Kapitel 5.2 vorgestellten Pattern-Sprache ist das Muster „Reiseticket kaufen“ enthalten, das an dieser Stelle verwendet wird.

Die Anwendung eines Patterns kann auf verschiedene Arten erfolgen. In jedem Fall muss zunächst die Stelle, d. h. diejenige Aufgabe im Aufgaben-Modell festgelegt bzw. markiert werden, bei der das im Muster vorhandene Task-Modell-Fragment (TMF) eingefügt werden soll. Hierfür stehen folgende drei Optionen zur Verfügung:

1. Die markierte Aufgabe bleibt erhalten und das Wurzel-Element des TMF wird als deren Nachfolger (rechter Nachbar) eingefügt.
2. Die markierte Aufgabe bleibt erhalten und das Wurzel-Element das TMF wird als deren Vorgänger (linker Nachbar) eingefügt.
3. Die markierte Aufgabe wird durch das Wurzel-Element des TMF ersetzt.

Das komplette TMF aus dem Muster wird an der entsprechenden Stelle in das Aufgaben-Modell kopiert. Die Verlinkung des Wurzel-Elements mit seinen Nachbar-Aufgaben erfolgt jeweils so, wie dies bereits in Kapitel 4.4.4.2 aufgezeigt wurde. Bei der dritten Option wird die Verlinkung der markierten Aufgabe in das Wurzel-Element des TMF übernommen, bevor sie aus dem Modell gelöscht wird.

Bei den folgenden Erklärungen wird stets davon ausgegangen, dass die Integration des TMF in das Task-Modell durch das bei der dritten Option beschriebene Verfahren erfolgt. Das Ersetzen der ursprünglichen Aufgabe „Reiseticket kaufen“ im Task-Modell durch das TMF des Musters „Reiseticket kaufen“ wird in Abbildung 73 veranschaulicht. Hierbei ist das bisherige Aufgaben-Modell oben links, das TMF unten links und das entsprechend erweiterte Task-Modell auf der rechten Seite dargestellt.

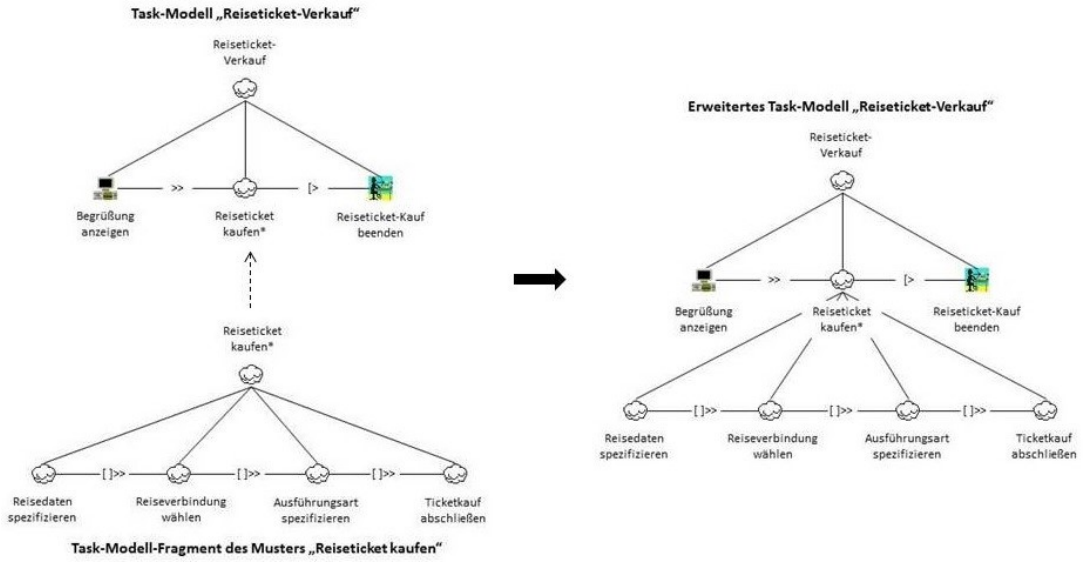


Abbildung 73 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDUNG DES MUSTERS „REISETICKET KAUFEN“

Das Muster „Reiseticket kaufen“ besitzt neben dem TMF auch ein Konzept-Modell-Fragment (CMF). Dessen Inhalt wird einfach an das Ende des bisherigen Konzept-Modells (siehe Tabelle 28) kopiert. In diesem Fall handelt es sich um ein einzelnes Konzept, das zur Ausführung der Aufgabe „Reiseticket kaufen“ benötigt wird und zum Starten des Ticketkauf-Vorgangs sorgt. In Tabelle 29 wird die derart erweiterte Konzept-Liste gezeigt.

Tabelle 29 ÜBERSICHT ÜBER DIE VERWENDETEN KONZEPTE (NACH ANWENDUNG DES 1. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
1	Begrüßung	DataOutput	Begrüßung anzeigen
2	Ticketkauf beenden	Activator	Reiseticket-Kauf beenden
3	Ticketkauf starten	Navigator	Reiseticket kaufen

Die vollständige Spezifikation des Patterns einschließlich seiner Modell-Fragmente kann in Anhang J.1 eingesehen werden.

Der aktuelle Stand des Aufgaben-Modells (siehe Abbildung 73 rechts) dient nun als Basis für die weitere Detaillierung. Zunächst wird die abstrakte Aufgabe „Reisedaten spezifizieren“

verfeinert. Auch hierfür steht wieder ein passendes Muster zur Verfügung. Das Auffinden von infrage kommenden Mustern wird durch die in der Pattern-Sprache modellierten Beziehungen wesentlich erleichtert.

In analoger Weise zum vorherigen Schritt wird nun das TMF aus dem Muster „Reisedaten spezifizieren“ in das Aufgaben-Modell integriert. Das Ergebnis wird in Abbildung 74 illustriert. Die beiden roten Umrandungen zeigen jeweils die TMF der Muster „Reiseticket kaufen“ und „Reisedaten spezifizieren“.

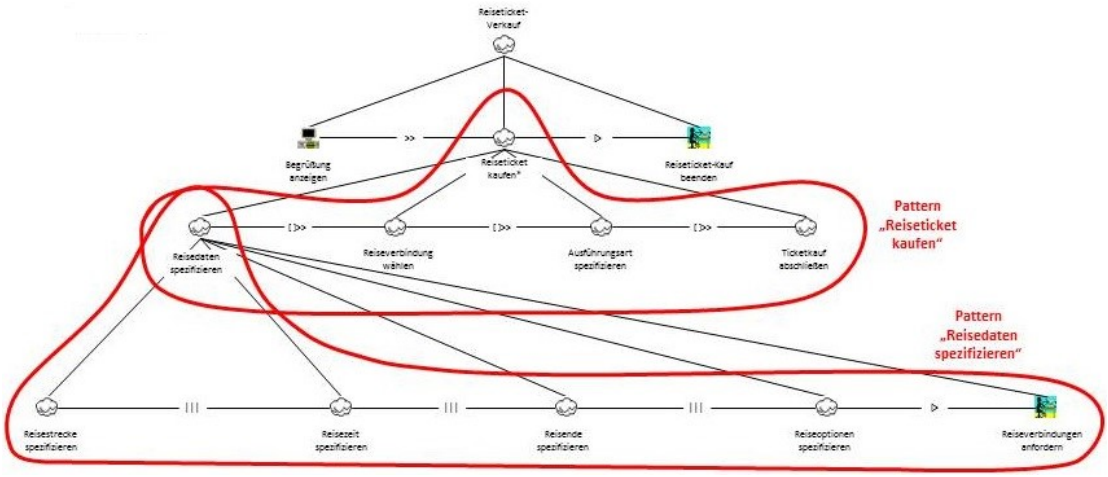


Abbildung 74 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDUNG DES MUSTERS „REISEDATEN SPEZIFIZIEREN“

Die Aufgabe „Reisedaten spezifizieren“ verfügt nun über Unteraufgaben zur Spezifikation der Reisestrecke, der Reisezeit, der Reisenden und weiteren, tarifrelevanten Reiseoptionen. Wurden alle geforderten Angaben gemacht, so können die Eingaben an die zugrundeliegende Geschäftslogik der Anwendung übergeben und die passenden Reiseverbindungen angefordert werden. Dies wird durch das Konzept „Reiseverbindungen anfordern“, das im CMF des Patterns „Reisedaten spezifizieren“ definiert ist, ermöglicht. Das Konzept wird, wie in Tabelle 30 gezeigt, an die Konzept-Liste angefügt.

Tabelle 30 ÜBERSICHT ÜBER DIE VERWENDETEN KONZEPTE (NACH ANWENDUNG DES 2. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
1	Begrüßung	DataOutput	Begrüßung anzeigen
2	Ticketkauf beenden	Activator	Reiseticket-Kauf beenden
3	Ticketkauf starten	Navigator	Reiseticket kaufen
4	Reiseverbindungen anfordern	Activator	Reiseverbindungen anfordern

Die Komplette, PPSL-konforme Spezifikation des Musters „Reisedaten spezifizieren“ befindet sich in Anhang J.2.

Nun geht es mit der Verfeinerung der abstrakten Aufgaben „Reisestrecke spezifizieren“, „Reisezeit spezifizieren“, „Reisende spezifizieren“ und „Reiseoptionen spezifizieren“ weiter. Hierfür kommen wieder entsprechende Patterns zum Einsatz, die grundsätzlich, wie bereits gezeigt, angewendet werden. Im Unterschied zu bisher verfügen die beiden Muster „Reisende spezifizieren“ und „Reiseoptionen spezifizieren“ jedoch über sogenannte Dummies, die eine konkrete Anpassung an den jeweiligen Einsatzkontext ermöglichen.

Erkennbar ist die Existenz der Dummies an dem PPSL-Beschreibungselement <TMF_IncludesDummy>, dem bei diesen Patterns der Wert „True“ zugewiesen ist.

Auf der linken Seite von Abbildung 75 ist die grafische Repräsentation des TMF des Musters „Reisende spezifizieren“ zu sehen. Es beinhaltet Unteraufgaben zur Angabe der Anzahl der reisenden Erwachsenen und mitreisenden Kinder, sowie eine Dummy-Aufgabe zur Modellierung weiterer Personengruppen oder Arten von mitzuführenden Gütern. Wie dies bereits in Kapitel 4.4.4.2 gezeigt wurde, muss der PaMGIS-Benutzer vor der Anwendung des Patterns die Dummy-Aufgabe interaktiv an den gewünschten Kontext anpassen und kann gegebenenfalls durch wiederholte Nutzung des Dummy das TMF erweitern. Soll der Dummy nicht genutzt werden, so wird dieser einfach aus dem TMF gelöscht. Die rechte Seite von Abbildung 75 zeigt oben eine mögliche Version des TMF für den Anwendungsfall des Verkaufs von Bahnfahrkarten und unten für Flugtickets. Hierbei können für eine Bahnfahrt optional die Anzahl mitgenommener Fahrräder und Hunde angegeben werden, für Flüge wurde beispielhaft die optionale Eingabe von zusätzlichen Gepäckstücken modelliert. Optionale Aufgaben werden mit PaMGIS wie bei CTT mit eckigen Klammern gekennzeichnet.

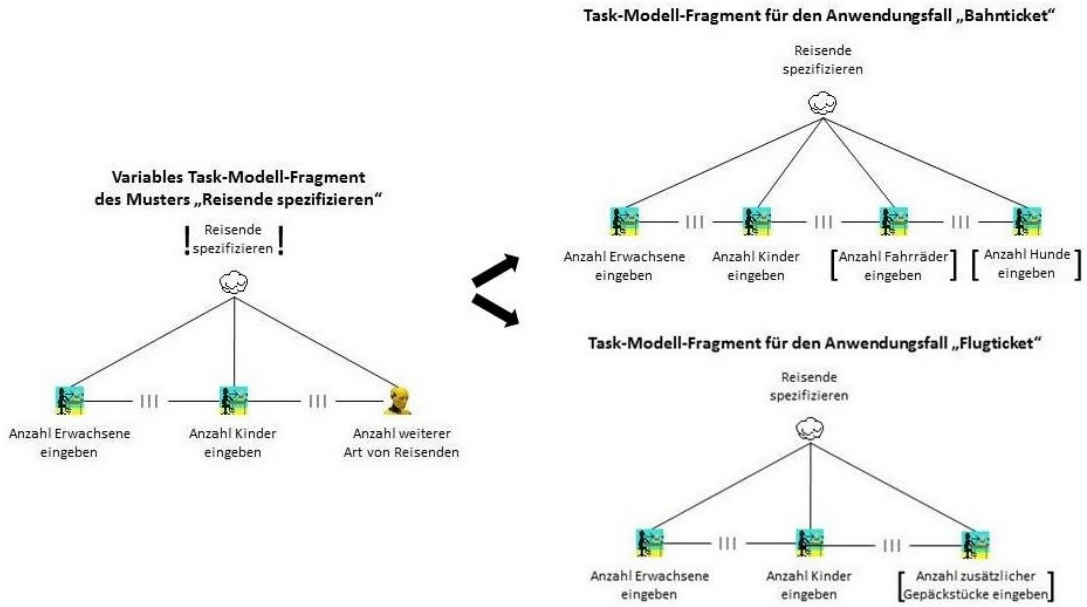


Abbildung 75 ANPASSEN DES VARIABLEN TMF DES MUSTERS „REISENDE SPEZIFIZIEREN“ FÜR DIE ANWENDUNGSFÄLLE „BAHNTICKET“ UND „FLUGTICKET“

In analoger Weise wird in Abbildung 76 rechts das TMF des Musters „Reiseoptionen spezifizieren“ mit den Dummy-Aufgaben und links die in der Fallstudie verwendete Version für den Bahnverkehr veranschaulicht.

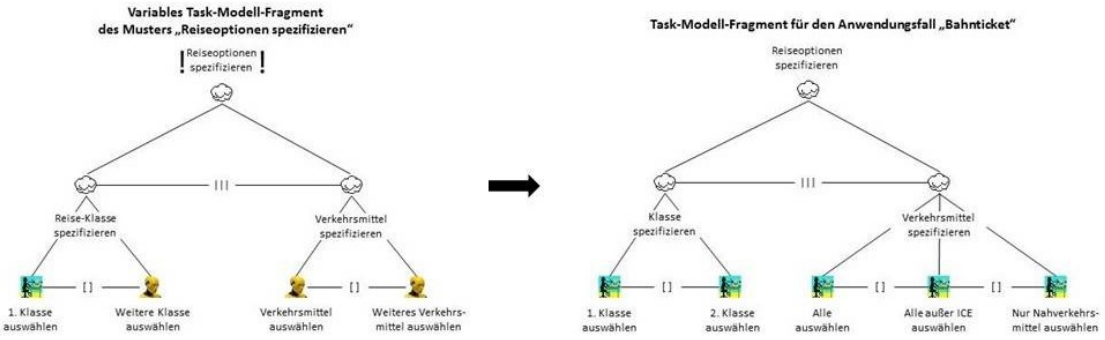


Abbildung 76 ANPASSEN DES TMF DES MUSTERS „REISEOPTIONEN SPEZIFIZIEREN“ FÜR DEN KONTEXT „BAHNTICKET“

Das durch die Anwendung der oben genannten vier Patterns resultierende Aufgaben-Modell ist in Abbildung 77 ersichtlich.

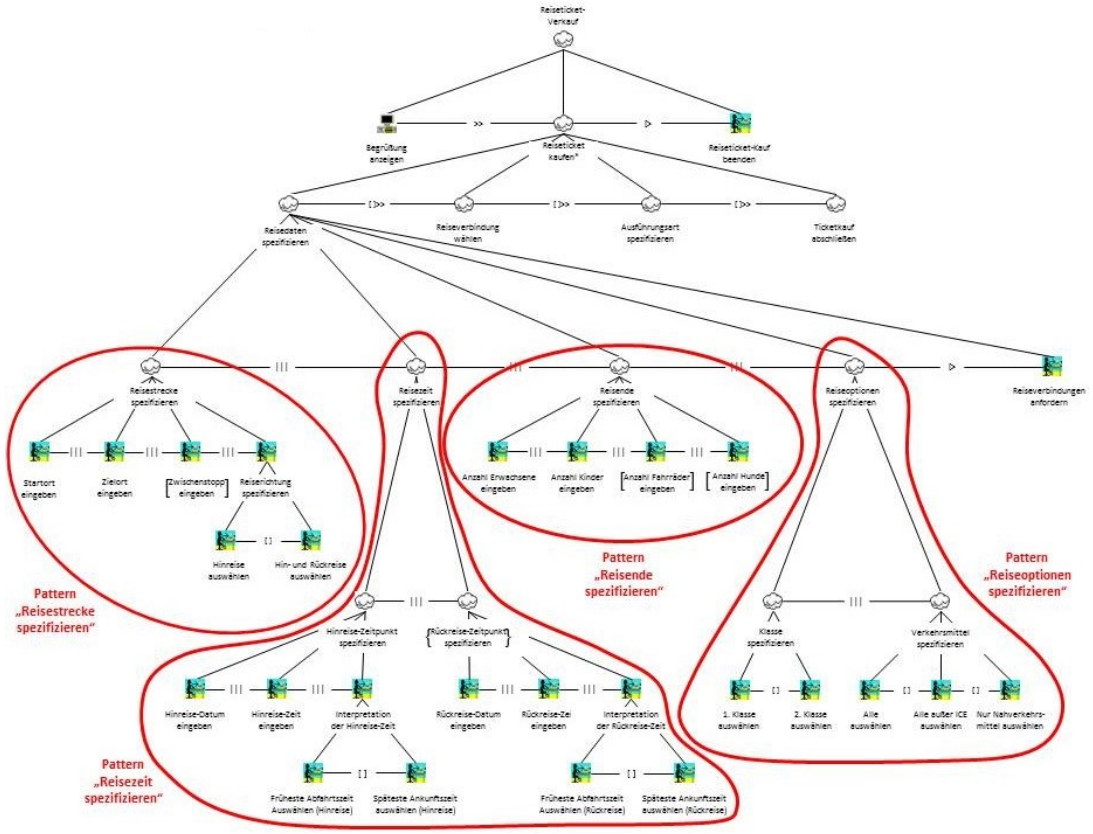


Abbildung 77 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDEN VIER WEITERE MUSTER

Bei der Aufgabe „Rückreise-Verbindung spezifizieren“ handelt es sich um eine bedingt auszuführende Aufgabe. Hierbei besitzt das Beschreibungselement <Conditional> der entsprechenden Aufgabe den Wert „True“. Die Bedingung, anhand der darüber entschieden wird, ob die Aufgabe ausgeführt wird oder nicht, ist im Beschreibungselement <Precondition> hinterlegt. Im vorliegenden Fall wird die Aufgabe nur dann ausgeführt, wenn vorher bei der Aufgabe „Reiserichtung spezifizieren“ die Option „Hin- und Rückreise“ ausgewählt wurde. Bedingt auszuführende Aufgaben werden mit geschweiften Klammern gekennzeichnet.

Aus Platzgründen befinden sich in Tabelle 31 nur die Konzepte, die durch die Anwendung der oben gezeigten Muster zu der Gesamtliste hinzukommen.

Tabelle 31 ÜBERSICHT ÜBER DIE ZUSÄTZLICHEN KONZEPTE (NACH ANWENDUNG DES 6. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
5	Startort	DataEdit	Startort eingeben
6	Zielort	DataEdit	Zielort eingeben
7	Zwischenstopp	DataEdit	Zwischenstopp eingeben
8	Reiserichtung	SingleChoice	Reiserichtung spezifizieren
9	Hinreise	ChoiceItem	Hinreise auswählen
10	Hin- und Rückreise	ChoiceItem	Hin- und Rückreise auswählen
11	Hinreisedatum	DataEdit	Hinreise-Datum eingeben
12	Hinreisezeit	DataEdit	Hinreise-Zeit eingeben
13	Interpretation Hinreisezeit	SingleChoice	Interpretation der Hinreise-Zeit
14	Hinreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Hinreise)
15	Hinreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Hinreise)
16	Rückreisedatum	DataEdit	Rückreise-Datum eingeben
17	Rückreisezeit	DataEdit	Rückreise-Zeit eingeben
18	Interpretation Rückreisezeit	SingleChoice	Interpretation der Rückreise-Zeit
19	Rückreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Rückreise)
20	Rückreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Rückreise)
21	Anzahl Erwachsene	DataEdit	Anzahl Erwachsene eingeben
22	Anzahl Kinder	DataEdit	Anzahl Kinder eingeben
23	Anzahl Fahrräder	DataEdit	Anzahl Fahrräder eingeben
24	Anzahl Hunde	DataEdit	Anzahl Hunde eingeben
25	Reiseklasse	SingleChoice	Reise-Klasse spezifizieren
26	Reiseklasse 1	ChoiceItem	1. Klasse auswählen
27	Reiseklasse 2	ChoiceItem	2. Klasse auswählen
28	Verkehrsmittel	SingleChoice	Verkehrsmittel spezifizieren
29	Verkehrsmittel 1	ChoiceItem	Alle auswählen
30	Verkehrsmittel 2	ChoiceItem	Alle außer ICE auswählen
31	Verkehrsmittel 3	ChoiceItem	Nur Nahverkehrsmittel auswählen

Es ist zu beachten, dass die Konzepte mit den laufenden Nummern 23, 24 und 27 sowie 29 bis 31 aus den zu den Dummy-Aufgaben korrespondierenden Dummy-Konzepten entstanden sind.

Die vollständigen Spezifikationen der verwendeten vier Muster befinden sich in Anhang J.3 bis J.6.

Nachdem die Modellierung des linken Teilbaums ab der Aufgabe „Reisedaten spezifizieren“ abgeschlossen ist, kann mit der Aufgabe „Reiseverbindung wählen“ fortgefahren werden. Hierbei kommt das gleichnamige Pattern zur Anwendung. Die Teilansicht des entsprechend

erweiterten Task-Modells ist in Abbildung 78 dargestellt. Ebenfalls wegen des hohen Platzbedarfs und der besseren Nachvollziehbarkeit erfolgt auch die Beschreibung bzw. Visualisierung des Aufgaben-Modells nur in Teilen.

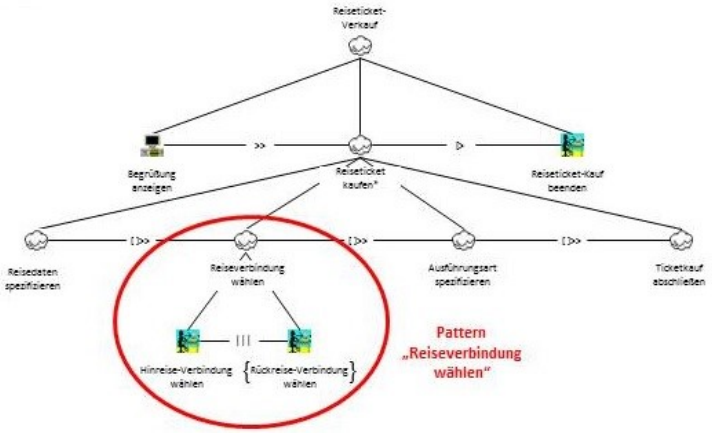


Abbildung 78 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDEN DES MUSTERS „REISEVERBINDUNG WÄHLEN“

Das Pattern verfügt über ein CMF mit zwei Konzepten zur Auswahl der Hinreise- und, falls bei der Angabe der Reisedaten eine Rückreise gewünscht wurde, zur Festlegung der Rückreiseverbindung. Diese Konzepte sind in Tabelle 32 aufgelistet.

Tabelle 32 ÜBERSICHT ÜBER DIE ZUSÄTZLICHEN KONZEPTE (NACH ANWENDUNG DES 7. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
32	Hinreiseverbindung	SingleChoice	Hinreise-Verbindung wählen
33	Rückreiseverbindung	SingleChoice	Rückreise-Verbindung wählen

Die Spezifikation des Patterns „Reiseverbindung wählen“ kann in Anhang J.7 eingesehen werden.

Weiter geht es mit der Detaillierung der abstrakten Aufgabe „Ausführungsart spezifizieren“. Das gleichnamige Muster liefert im TMF zwei abstrakte Unteraufgaben zur Festlegung der Zahlungsart und der Versandart für die Tickets. Die TMF der beiden hierfür zur Verfügung gestellten Patterns sind wiederum mit Dummies zur Anpassung an den vorliegenden Kontext ausgestattet.

Wie in Abbildung 79 links oben gezeigt wird, verfügt das TMF des Patterns „Zahlungsart spezifizieren“ über zwei Unteraufgaben; eine zur Auswahl der Zahlung per Kreditkarte und eine Dummy-Aufgabe zur Festlegung weiterer Zahlungsarten. In Rahmen dieser Fallstudie wird der Dummy genutzt, um zusätzlich die Zahlung per EC-Karte, Kundenkarte und Banküberweisung zu modellieren. Das derart angepasste TMF ist in der Abbildung oben rechts zu sehen. Auch das Muster „Versandart spezifizieren“ bringt zwei Unteraufgaben in seinem TMF mit. Erstere dient zur Auswahl des Ticketdrucks auf einem lokalen Drucker, bei der anderen handelt es sich wieder um einen Dummy zur Definition weiterer Versandmöglichkeiten. Hiermit wird im vorliegenden Beispiel zusätzlich der Versand per Email und SMS angeboten. Das zum Pattern gehörende TMF und dessen angepasste Ausprägung sind im unteren Teil von Abbildung 79 dargestellt.

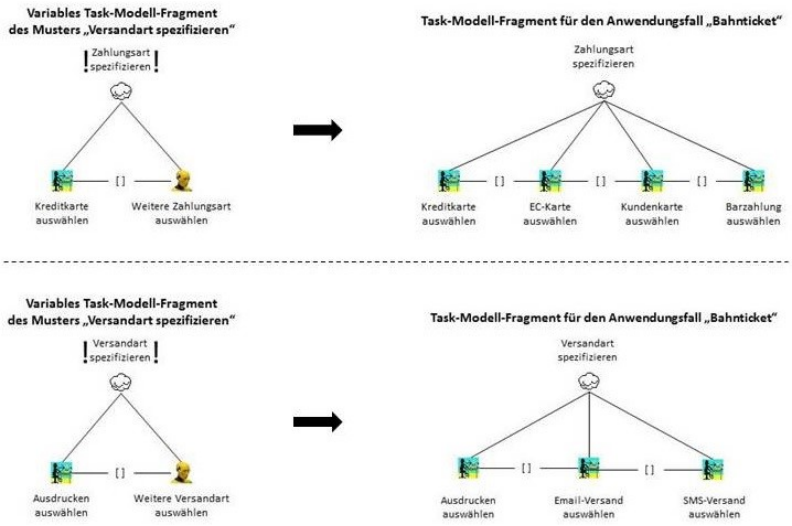


Abbildung 79 ANPASSEN DER TMF DER MUSTER „VERSANDART SPEZIFIZIEREN“ UND „ZAHLUNGSART SPEZIFIZIEREN“ FÜR DEN KONTEXT „BAHNTICKET“

Nach Abschluss der Modellierung des Teilbaums für die Aufgabe „Ausführungsart spezifizieren“ mithilfe der drei genannten Muster, stellt sich die Ansicht des hierfür relevanten Teils des Task-Modells wie in Abbildung 80 gezeigt dar.

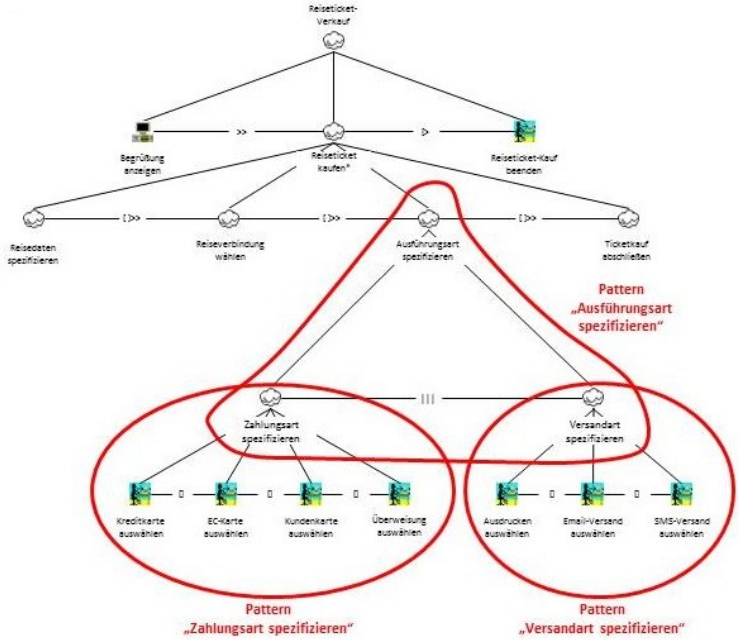


Abbildung 80 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDEN DER MUSTER „AUSFÜHRUNGSART SPEZIFIZIEREN“, „ZAHLUNGSART SPEZIFIZIEREN“ UND „VERSANDART SPEZIFIZIEREN“

Die beiden Patterns „Zahlungsart spezifizieren“ und „Versandart spezifizieren“ umfassen jeweils ein CMF mit den benötigten Konzepten. Diese sind in Tabelle 33 aufgelistet.

Tabelle 33 ÜBERSICHT ÜBER DIE ZUSÄTZLICHEN KONZEPTE (NACH ANWENDUNG DES 10. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
34	Zahlungsart	SingleChoice	Zahlungsart spezifizieren
35	Zahlungsart 1	ChoiceItem	Kreditkarte auswählen
36	Zahlungsart 2	ChoiceItem	EC-Karte auswählen
37	Zahlungsart 3	ChoiceItem	Kundenkarte auswählen
38	Zahlungsart 4	ChoiceItem	Überweisung auswählen
39	Versandart	SingleChoice	Versandart spezifizieren
40	Versandart 1	ChoiceItem	Ausdrucken auswählen
41	Versandart 2	ChoiceItem	Email-Versand auswählen
42	Versandart 3	ChoiceItem	SMS-Versand auswählen

Die Konzepte mit den laufenden Nummern 36 bis 38 sowie 41 und 42 sind aus den zu den Dummy-Aufgaben korrespondierenden Dummy-Konzepten entstanden.

Die PPSL-Repräsentation der drei in Abbildung 80 gekennzeichneten Patterns sind in Anhang J.8 bis J.10 zu finden.

Um das Domänen-Modell fertig zu stellen, muss schließlich noch die abstrakte Aufgabe „Ticketkauf abschließen“ detailliert werden. Dies geschieht unter Verwendung der beiden Patterns „Ticketkauf abschließen“ und „Eingaben überprüfen“.

Das Abschließen des Ticketkaufs erfolgt durch zwei Arbeitsschritte. Zunächst sind die relevanten Eingaben und Auswahlen vom Benutzer nochmals zu überprüfen und dann muss der rechtsverbindliche Kaufauftrag bestätigt bzw. an die zugrundeliegende Geschäftsanwendung gesendet werden. Das TMF des Musters zur Überprüfung der gemachten Eingaben besteht im Wesentlichen aus zwei Dummy-Tasks, mit deren Hilfe festgelegt werden kann, welche der gemachten Angaben dem Benutzer zur nochmaligen Überprüfung vorgelegt werden sollen. Wie aus Abbildung 81 entnommen werden kann, werden in der Fallstudie die gewählte Hinreise- und gegebenenfalls die Rückreiseverbindung sowie die gewünschten Zahlungs- und Versandarten zum Review angezeigt.

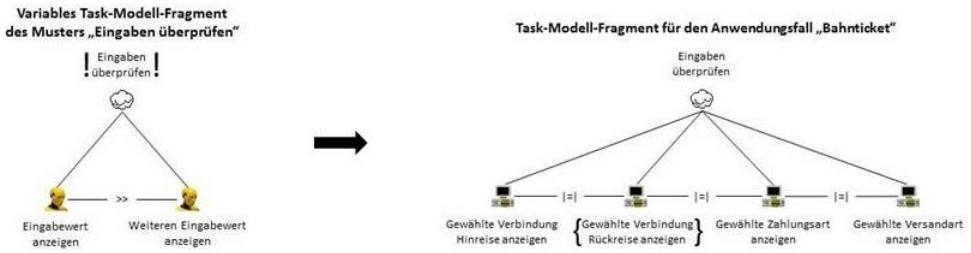


Abbildung 81 ANPASSEN DES TMF DES MUSTERS „EINGABEN ÜBERPRÜFEN“ FÜR DEN KONTEXT „BAHNTICKET“

Der betreffende Teil des Task-Modells nach der Anwendung der beiden Patterns ist in Abbildung 82 skizziert.

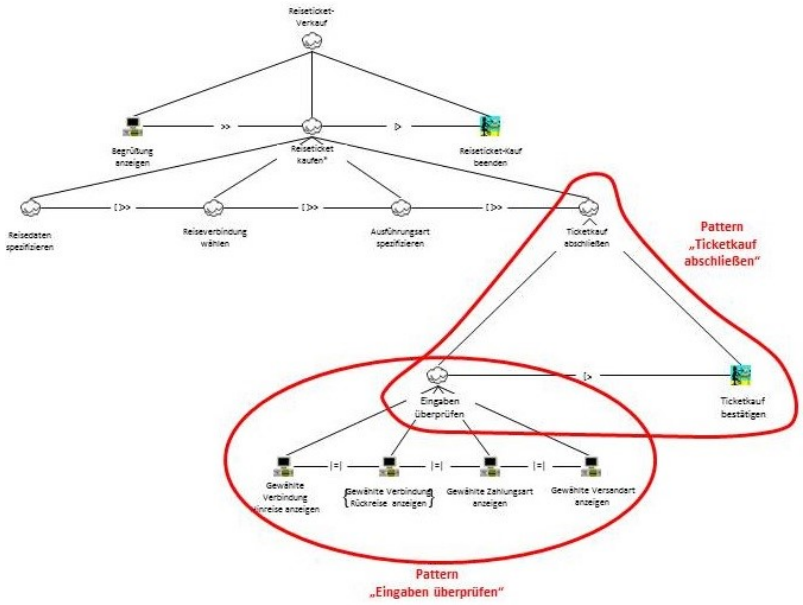


Abbildung 82 ERWEITERN DES AUFGABEN-MODELLS DURCH ANWENDEN DER MUSTER „TICKETKAUF ABSCHLIEßEN" UND „EINGABEN ÜBERPRÜFEN"

Die Konzepte aus den CMF der beiden Muster sind in Tabelle 34 ersichtlich. Die Konzepte mit den laufenden Nummern 44 bis 47 sind von den Dummies abgeleitet.

Tabelle 34 ÜBERSICHT ÜBER DIE ZUSÄTZLICHEN KONZEPTE (NACH ANWENDUNG DES 12. MUSTERS)

Nr.	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
43	Ticketkauf bestätigen	Activator	Ticketkauf bestätigen
44	Eingabewert 1	DataOutput	Gewählte Verbindung Hinreise anzeigen
45	Eingabewert 2	DataOutput	Gewählte Verbindung Rückreise anzeigen
46	Eingabewert 3	DataOutput	Gewählte Zahlungsart anzeigen
47	Eingabewert 4	DataOutput	Gewählte Versandart anzeigen

Die PPSL-konformen Spezifikation der Patterns „Ticketverkauf abschließen" und „Eingaben überprüfen" sind in Anhang J.11 und J.12 zu finden.

5.5. Erstellung der Dialog-Modelle

In diesem Kapitel erfolgt die Entwicklung der Dialog-Modelle für die beiden in Kapitel 5.3 definierten Benutzungskontexte. Diese unterscheiden sich hinsichtlich des jeweils zum Einsatz kommenden Endgeräts. Es handelt sich einerseits um den Kontext „Großer Bildschirm" mit einem Standard-Desktop-PC und andererseits um den Kontext „Kleiner Bildschirm" mit einem Standard-Smartphone (siehe Abbildung 65 und Abbildung 66 auf Seite 174).

Die Erstellung des Domänen-Modells wurde mit dem in Abbildung 72 dargestellten, initialen Aufgaben-Modell begonnen. Hierfür bildet ein entsprechendes Dialog-Modell den jeweiligen

Ausgangspunkt für die kontextabhängige Modellierung der Dialoge. Dieses ist in Abbildung 83 zu sehen.

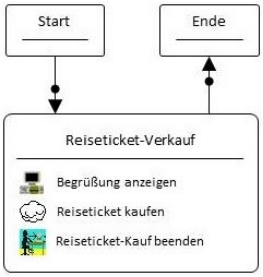


Abbildung 83 ANFÄNGLICHES DIALOG-MODELL

Wie bei der Konstruktion des Domänen-Modells können auch bei der Erstellung der Dialog-Modelle die Patterns wertvolle Hilfe leisten. In diesem Fall werden die Dialog-Modell-Fragmente (DMF) genutzt. Analog zum Vorgehen beim Task- und Konzept-Modell wird auch hier mit dem Muster „Reiseticket kaufen“ begonnen. Dieses Pattern verfügt nur über ein einziges DMF, das für beide Benutzungskontexte verwendet werden kann. Es ist in Abbildung 84 schematisch dargestellt.

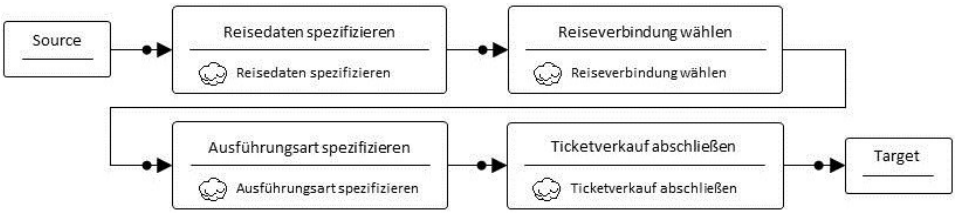


Abbildung 84 DMF DES MUSTERS „REISETICKET KAUFEN“

Der Beginn eines DMF wird jeweils durch einen Platzhalter namens „Source“ (engl. Quelle) und das Ende durch „Target“ (engl. Ziel) gekennzeichnet. Bei der Integration des DMF in das Dialog-Modell wird in diesem Schritt für Quelle und Ziel jeweils der Dialog „Reiseticket-Verkauf“ entsprechend verlinkt. Dieser wird hierfür beim ersten Dialog des DMF in das Beschreibungselement <Predecessor> und beim letzten als <Successor> eingetragen. Das Resultat ist in Abbildung 85 zu sehen.

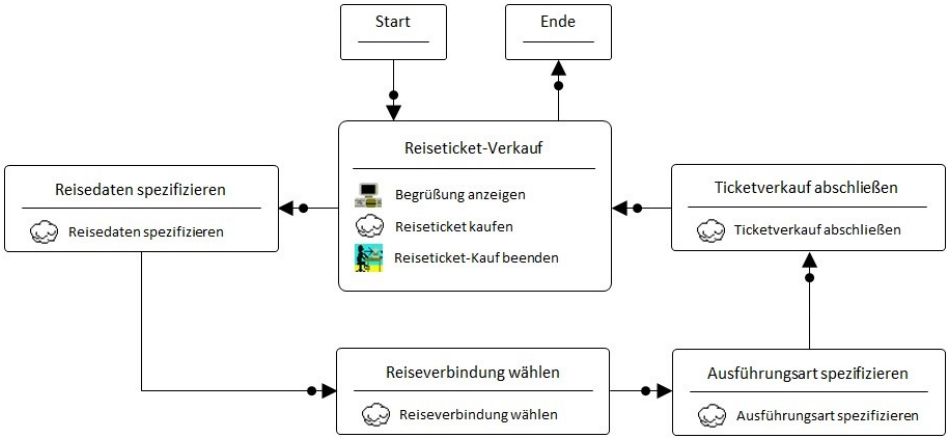


Abbildung 85 DIALOG-MODELL ERWEITERT UM DAS DMF DES MUSTERS „REISETICKET KAUFEN“

Bis zu diesem Stand ist das Dialog-Modell für beide in dieser Fallstudie berücksichtigten Benutzungskontexte gleich. Es folgt nun zunächst die Beschreibung der Dialogmodellierung für den Kontext „Großer Bildschirm“.

5.5.1. Dialog-Modell für Kontext „Großer Bildschirm“

Das Muster „Reisedaten spezifizieren“ besitzt für jeden der beiden Kontexte unterschiedliche DMF. Das DMF für große Bildschirme fasst die entsprechenden Aufgaben des TMF, wie in Abbildung 86 illustriert, in einem einzigen Dialog zusammen.

Abbildung 86 DMF des Musters „Reisedaten spezifizieren“ für große Bildschirme

Für die Integration eines DMF in ein Dialog-Modell gibt es drei grundsätzliche Möglichkeiten:

1. Das DMF wird nach einem bestehenden Dialog eingefügt, so wie dies beispielsweise beim ersten Schritt durchgeführt wurde (siehe Abbildung 85 auf Seite 187).
2. Ein bestehender Dialog wird durch das DMF ersetzt.
3. Der Inhalt eines oder mehrerer Dialoge des DMF wird in einen bestehenden Dialog des Dialog-Modells eingefügt.

Bei den ersten beiden Optionen müssen jeweils die Vorgänger- und Nachfolger-Dialoge des DMF entsprechend angepasst werden.

Hier wird nun der bestehende Dialog „Reisedaten spezifizieren“ durch den Dialog aus dem DMF ersetzt (Option 2). Das Ergebnis ist in Abbildung 87 dargestellt.

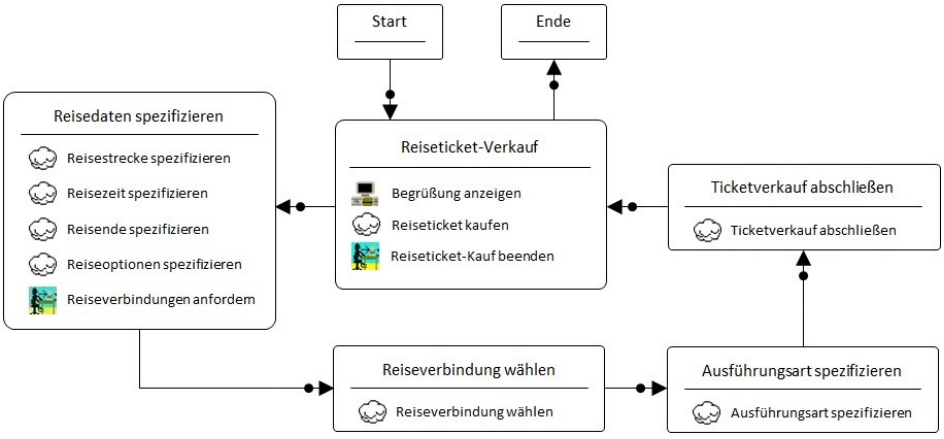


Abbildung 87 Dialog-Modell für große Bildschirme erweitert um das entsprechende DMF des Musters „Reisedaten spezifizieren“

Als nächstes kommt das Pattern „Reisestrecke spezifizieren“ an die Reihe. Es verfügt über ein einzelnes DMF. Dieses beinhaltet, wie in Abbildung 88 gezeigt, die vier Interaktions-Aufgaben des korrespondierenden TMF.

Abbildung 88 DMF DES MUSTERS „REISESTRECKE SPEZIFIZIEREN“

Die Aufgaben im Dialog werden durch das unter der dritten Möglichkeit beschriebenen Verfahren in den Dialog „Reisedaten spezifizieren“ integriert. Es ist zu beachten, dass die abstrakte Aufgabe „Reisestrecke spezifizieren“ dabei erhalten bleibt. Sie kann bei der späteren Ableitung der Benutzeroberflächenmodelle als Gruppierungsmerkmal dienen. Das derart erweiterte Dialog-Modell ist in Abbildung 89 illustriert.

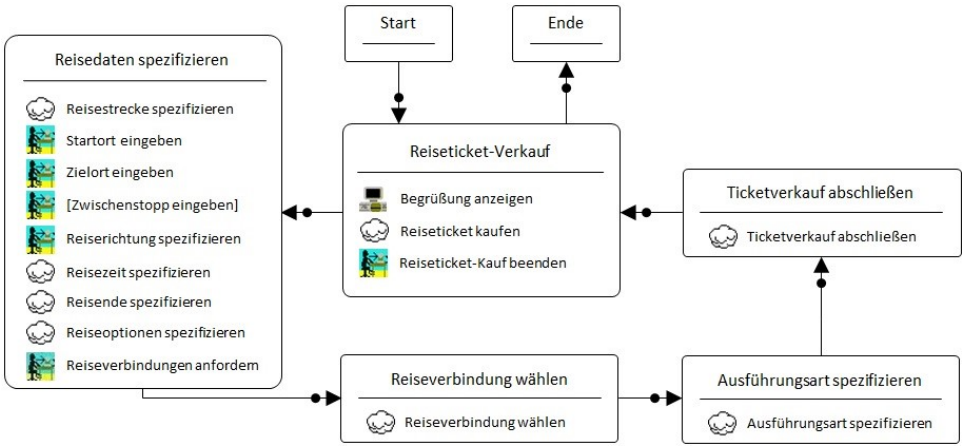


Abbildung 89 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DAS DMF DES MUSTERS „REISESTRECKE SPEZIFIZIEREN“

Auch das Pattern „Reisezeit spezifizieren“ besitzt nur ein DMF mit einem einzelnen Dialog. Wie Abbildung 90 entnommen werden kann, umfasst dieser die Aufgaben zur Eingabe der Reisezeitpunkte für die Hin- und gegebenenfalls auch für die Rückreise.

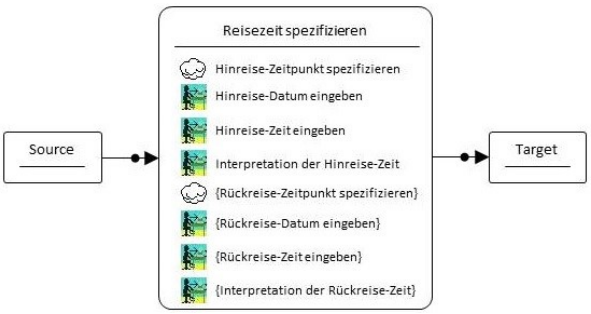


Abbildung 90 DMF DES MUSTERS „REISEZEIT SPEZIFIZIEREN“

Die Übernahme des DMF in das Dialog-Modell erfolgt wie im vorhergehenden Schritt gemäß der dritten Integrationsmöglichkeit. Auch hier werden die Aufgaben in den Dialog „Reisedaten spezifizieren“ eingefügt. Das Ergebnis ist in Abbildung 91 visualisiert.

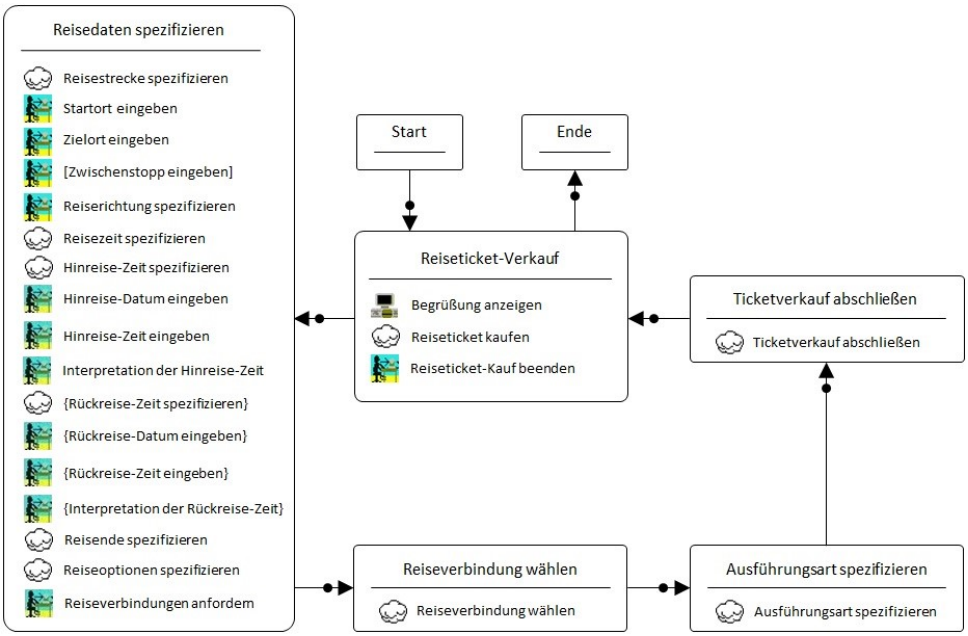


Abbildung 91 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DAS DMF DES MUSTERS „REISEZEIT SPEZIFIZIEREN“

Die Task-Modell-Fragmente der Muster „Reisende spezifizieren“ und „Reiseoptionen spezifizieren“ enthalten Dummy-Aufgaben. Diese wurden im Rahmen der Erstellung des Domänen-Modells an den entsprechenden Einsatzzweck angepasst (siehe Abbildung 75 auf Seite 180 und Abbildung 76 auf Seite 181). Da in den Spezifikationen der beiden DMF jeweils das Beschreibungselement <DLG_Processing> mit dem Wert „Recursive“ belegt ist, werden die durch den PaMGIS-Benutzer angepassten Aufgaben automatisch in die Dialog-Modell-Fragmente übernommen. In Abbildung 92 ist auf der linken Seite das endgültige DMF des Patterns „Reisende spezifizieren“ und auf der linken Seite das DMF des Musters „Reiseoptionen spezifizieren“ zu sehen.

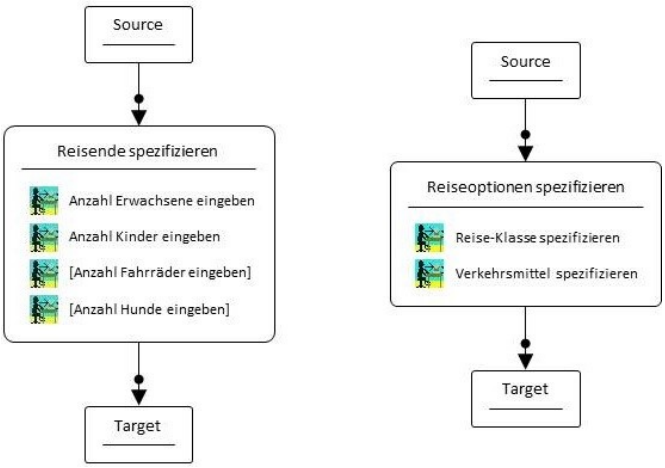


Abbildung 92 DMF DER MUSTER „REISENDE SPEZIFIZIEREN“ UND „REISEOPTIONEN SPEZIFIZIEREN“

Diese beiden DMF werden nach der dritten Integrationsmethode in den Dialog „Reisedaten spezifizieren“ eingefügt. Somit ist dieser Dialog fertig modelliert. Er wird in Abbildung 93 vollständig gezeigt.

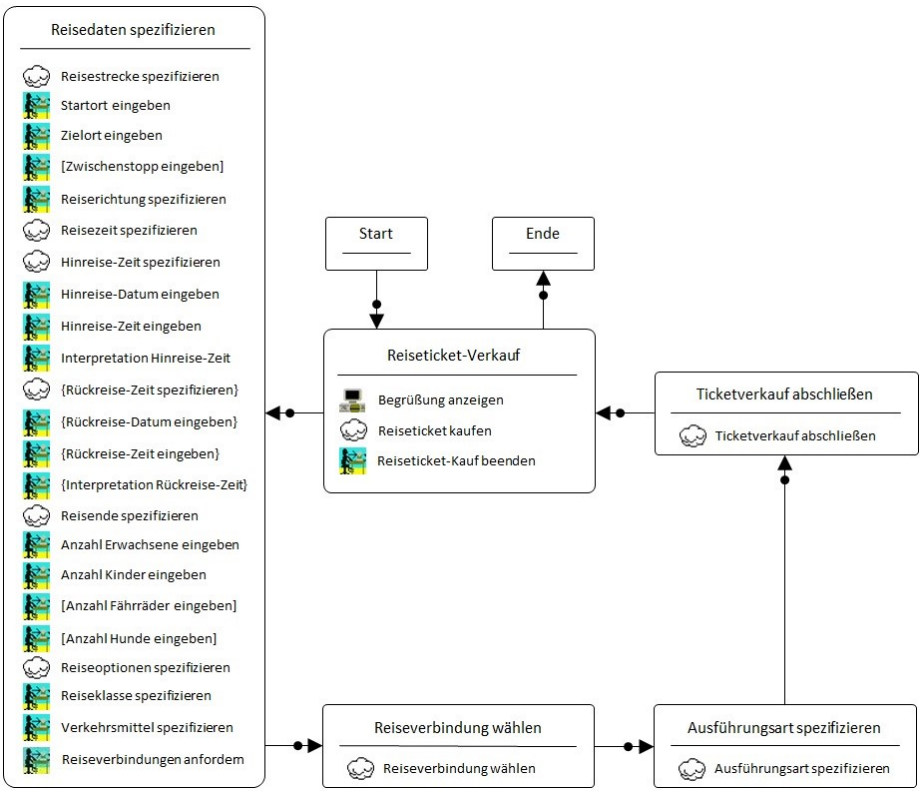


Abbildung 93 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DIE DMF DER MUSTER „REISENDE SPEZIFIZIEREN“ UND „REISEOPTIONEN SPEZIFIZIEREN“

Bei der Dialog-Modellierung bzw. der Spezifikation von Dialog-Modell-Fragmenten kann es vorkommen, dass zusätzliche Konzepte zur Navigation zwischen den Dialogen benötigt werden. Solche Konzepte besitzen aber keine korrespondierende Aufgabe im Task-Modell. Es ist deshalb auch nicht möglich, die Konzepte über den herkömmlichen Weg, d. h. durch das Einfügen einer Aufgabe, im Dialog zu berücksichtigen. Aus diesem Grund verfügen Dialog-Spezifikationen in PaMGIS über das Beschreibungselement <SupplementaryConcepts> mit dessen Hilfe derartige zusätzlichen Konzepte hinzugefügt werden können. Dieses Beschreibungselement beherbergt dann eine oder mehrere Referenzen auf im Konzept-Modell bzw. CMF definierte Konzepte.

Das in Abbildung 94 dargestellte DMF des Patterns „Reiseverbindung wählen“ für große Bildschirme verfügt über solch ein zusätzliches Navigator-Konzept, das im gleichnamigen Dialog als letztes Element aufgeführt ist.

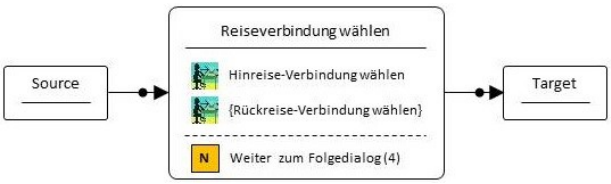


Abbildung 94 DMF DES MUSTERS „REISEVERBINDUNG WÄHLEN“ FÜR GROßE BILDSCHIRME

Der Name „Weiter zum Folgedialog (4)“ ergibt sich hierbei aus der Tatsache, dass bei der Erstellung des Dialog-Modells für kleine Bildschirme noch weitere zusätzliche Konzepte erforderlich sind (siehe Kapitel 5.5.2) und entsprechend durchnummeriert wurden.

Wie aus Abbildung 95 entnommen werden kann, wird das DMF durch Dialogersetzung (zweite Methode) in das Dialog-Modell integriert.

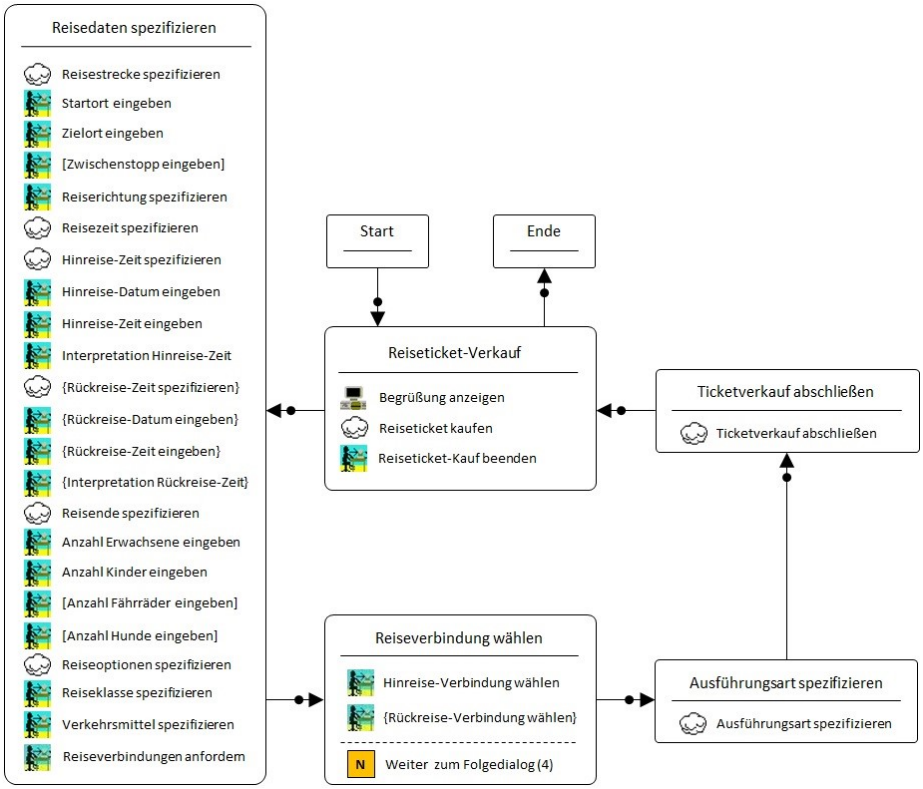


Abbildung 95 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DAS ENTSPRECHENDE DMF DES MUSTERS „REISEVERBINDUNG WÄHLEN“

Im folgenden Schritt kommt das DMF des Musters „Ausführungsart spezifizieren“ zum Einsatz. Auch dieses besitzt, wie in Abbildung 96 zu sehen, ein zusätzlichen Navigations-Konzept mit dem Namen „Weiter zum Folgedialog (6)“.

Abbildung 96 DMF DES MUSTERS „AUSFÜHRUNGSART SPEZIFIZIEREN“

Auch dieses DMF wird durch das Mittel der Dialogersetzung in das Dialog-Modell übernommen. Der neue Stand des Modells ist in Abbildung 97 illustriert.

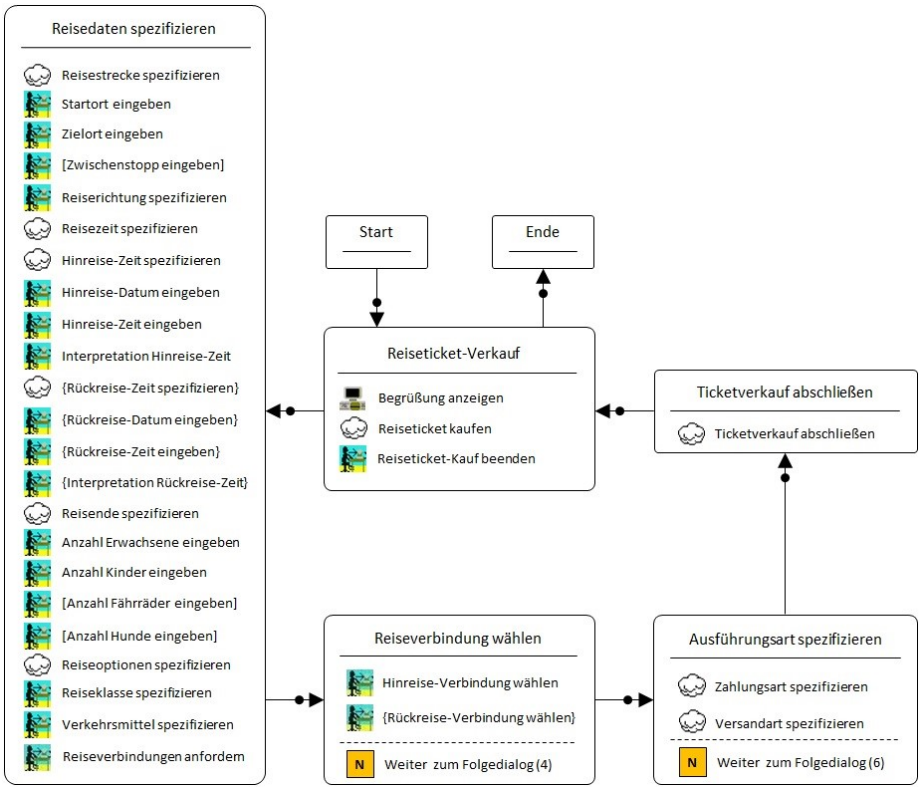


Abbildung 97 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DAS DMF DES MUSTERS „AUSFÜHRUNGSART SPEZIFIZIEREN“

Abbildung 98 zeigt auf der linken Seite das DMF des Patterns „Zahlungsart spezifizieren“ und rechts das DMF des Patterns „Versandart spezifizieren“.

Abbildung 98 DMF DER MUSTER „ZAHLUNGSART SPEZIFIZIEREN“ UND „VERSANDART SPEZIFIZIEREN“

Diese beiden DMF werden durch die dritte Integrationsmethode in den bestehenden Dialog namens „Ausführungsart spezifizieren“ aufgenommen. Das Ergebnis ist in Abbildung 99 visualisiert.

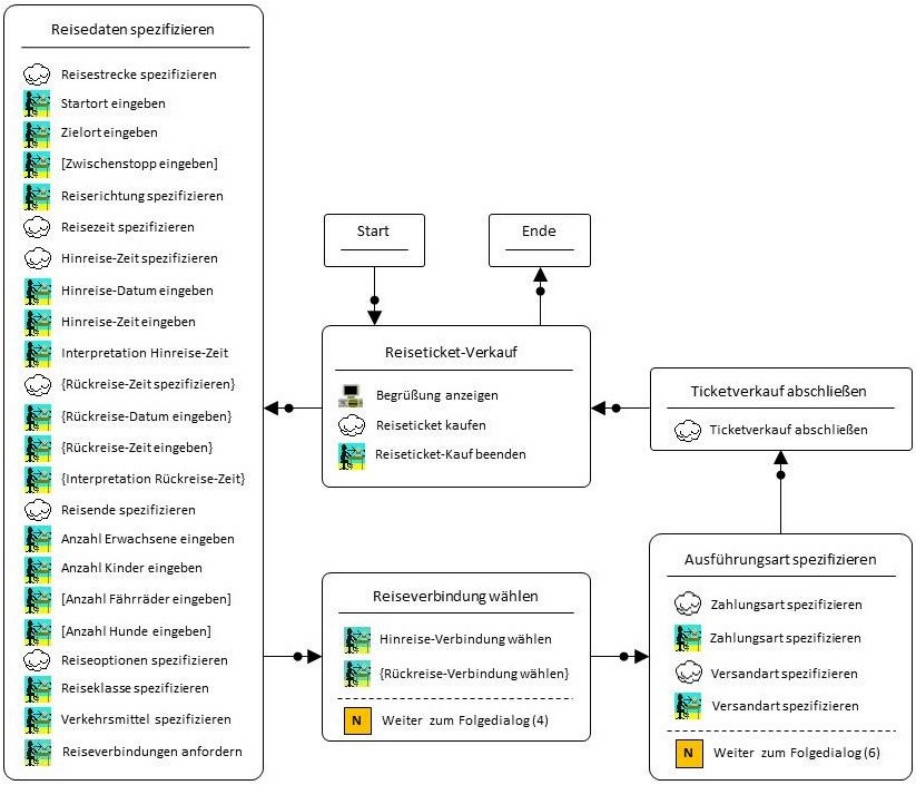


Abbildung 99 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DIE DMF DER MUSTER „ZAHLUNGSART SPEZIFIZIEREN“ UND „VERSANDART SPEZIFIZIEREN“

Danach muss noch das in Abbildung 100 dargestellte Dialog-Modell-Fragment des Musters „Ticketkauf abschließen“ in das Dialog-Modell eingebaut werden. Dies erfolgt durch die Ersetzung des gleichnamigen Dialogs gemäß des Verfahrens der zweiten Integrationsoption.

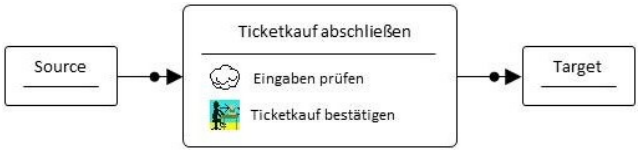


Abbildung 100 DMF DES MUSTERS „TICKETKAUF ABSCHLIEßEN“

Das Resultat dieser Dialog-Ersetzung kann in Abbildung 101 eingesehen werden.

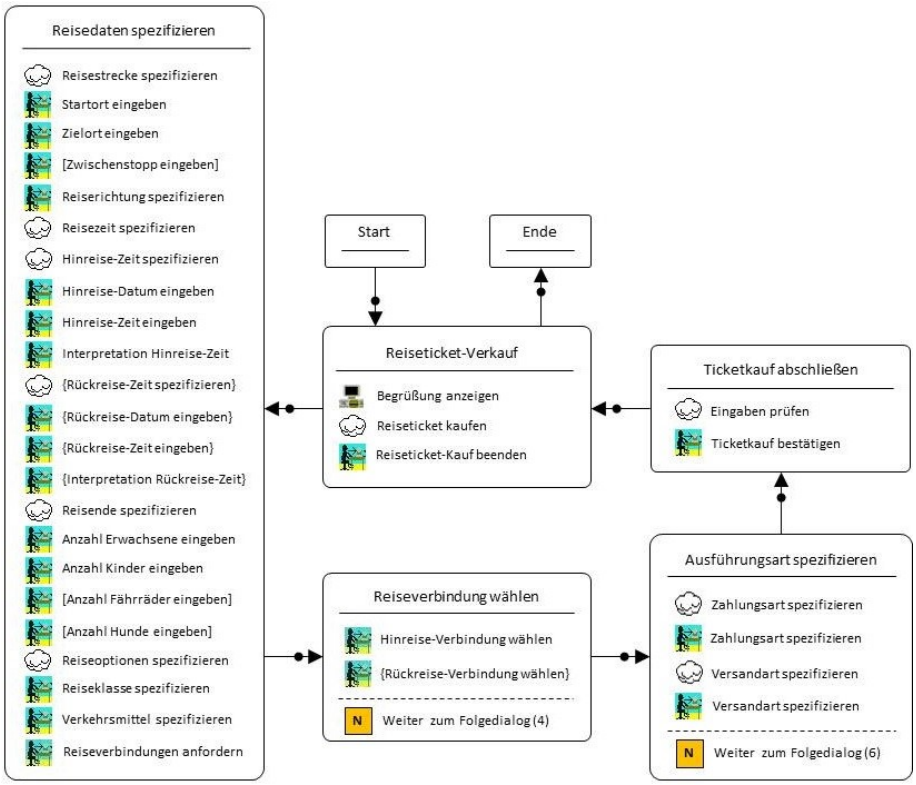


Abbildung 101 DIALOG-MODELL FÜR GROßE BILDSCHIRME ERWEITERT UM DAS DMF DES MUSTERS „TICKETKAUF ABSCHLIEßEN“

Schließlich kommt noch das Dialog-Modell-Fragment des Patterns „Eingaben überprüfen“ zum Einsatz.

Wie auch schon bei den Mustern „Reisende spezifizieren“ und „Reiseoptionen spezifizieren“ enthält auch hier das korrespondierende Task-Modell-Fragment Dummy-Aufgaben, das an den Einsatzzweck der Fallstudie adaptiert wurde (siehe Abbildung 81 auf Seite 185). Durch die Belegung des Beschreibungselements <DLG_Processing> mit dem Wert „Recursive“ werden wieder die vom PaMGIS-User vorgenommenen Anpassungen automatisch in das Dialog-Modell-Fragment übernommen. Das DMF in seiner endgültigen Form ist in Abbildung 102 zu sehen.

Abbildung 102 DMF DES MUSTERS „EINGABEN ÜBERPRÜFEN“

Zum Abschluss wird das fertige Dialog-Modell für einen Standard-Desktop-PC mit großem Bildschirm in Abbildung 103 gezeigt.

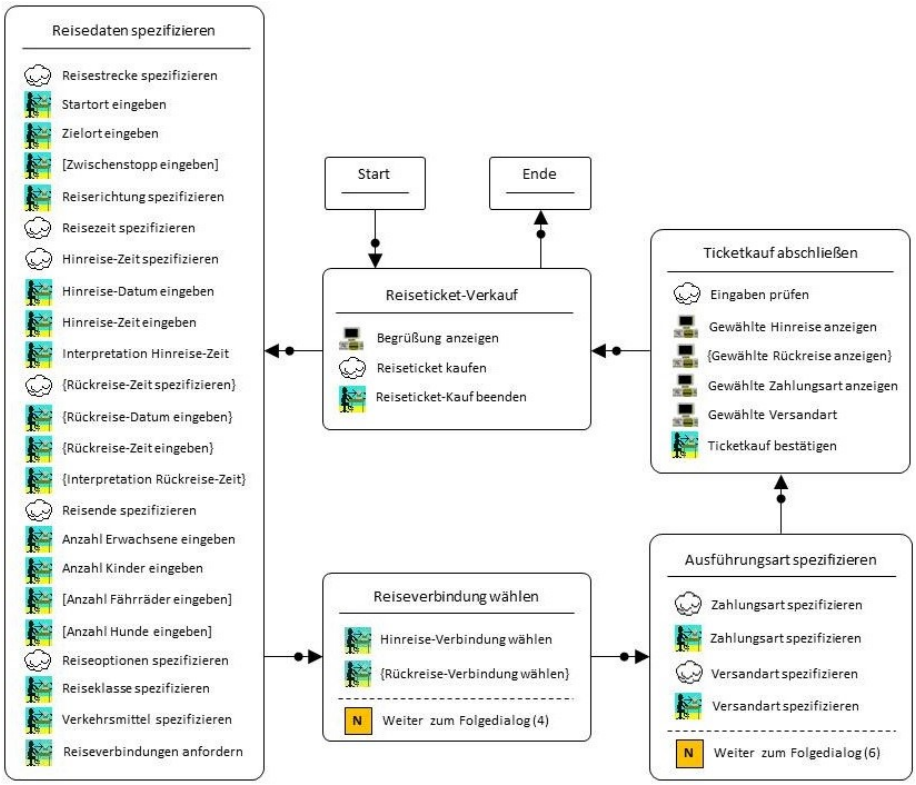


Abbildung 103 FERTIGES DIALOG-MODELL FÜR GROßE BILDSCHIRME NACH DER ERWEITERUNG UM DAS DMF DES MUSTERS „EINGABEN ÜBERPRÜFEN“

5.5.2. Dialog-Modell für den Kontext „Kleiner Bildschirm“

Ausgehend von dem in Abbildung 85 gezeigten Stand des Dialog-Modells, kommt bei dem Benutzungskontext mit Standard-Smart-Phone das DMF des Patterns „Reisedaten spezifizieren“ für kleine Bildschirme zum Einsatz. Bei diesem werden, im Unterschied zum DMF für große Bildschirme, die im TMF befindlichen Aufgaben nicht in einem einzigen, sondern in eine Sequenz von vier Dialogen untergebracht. Die Dialog-Struktur wurde hierbei auf Basis des Musters „Wizard“ gestaltet. Die PPSL-konforme Spezifikation dieses Patterns ist in Anhang J.13 zu finden.

Wie dies Abbildung 104 entnommen werden kann, kommen bei den ersten drei Dialogen wieder zusätzliche Konzepte, welche die Navigation von einem zum nächsten Dialog ermöglichen, zum Einsatz.

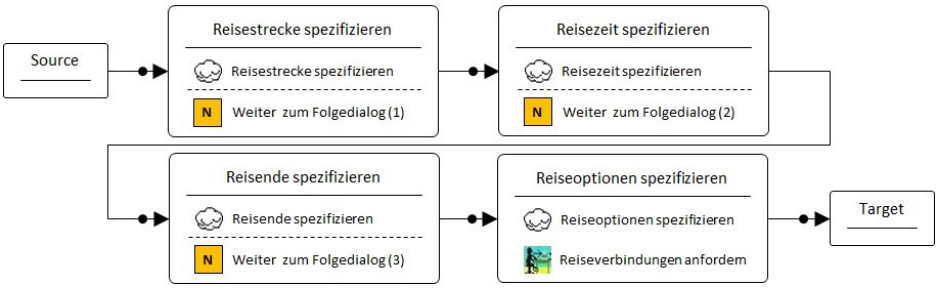


Abbildung 104 DMF DES MUSTERS „REISEDATEN SPEZIFIZIEREN“ FÜR KLEINE BILDSCHIRME

Bei der Integration ins Dialog-Modell wird der Dialog „Reisedaten spezifizieren“ durch die vier Dialoge des DMF ersetzt (Option 2). Das Ergebnis ist in Abbildung 105 visualisiert.

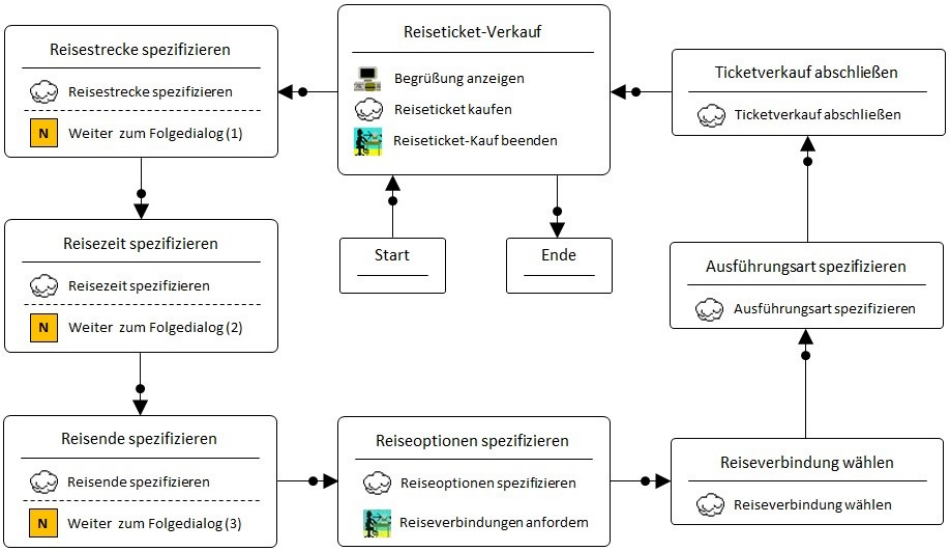


Abbildung 105 DIALOG-MODELL FÜR KLEINE BILDSCHIRME ERWEITERT UM DAS DMF DES MUSTERS „REISEDATEN SPEZIFIZIEREN“

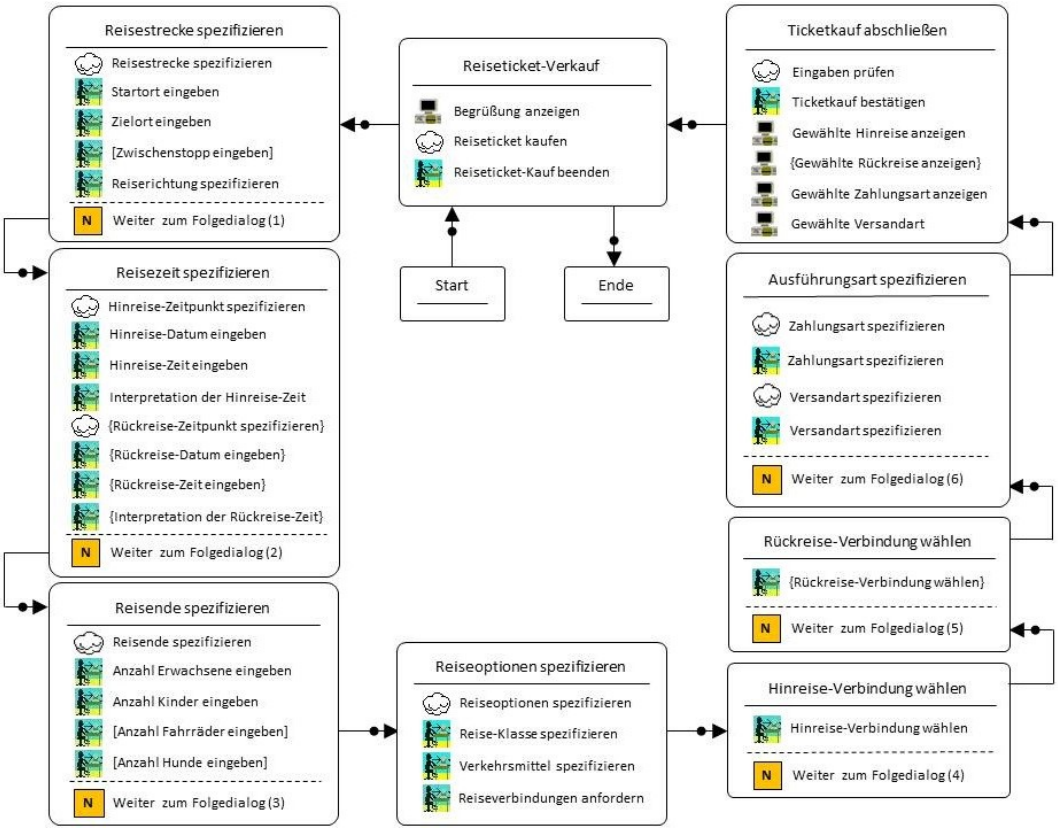


Abbildung 106 FERTIGES DIALOG-MODELL FÜR KLEINE BILDSCHIRME NACH DER ERWEITERUNG UM DAS DMF DES MUSTERS „EINGABEN ÜBERPRÜFEN“

Im weiteren Verlauf wird analog zur Erstellung des Dialog-Modells für große Bildschirme in Kapitel 5.5.1 schrittweise vorgegangen. Sofern verfügbar, kommen dabei natürlich die DMF für kleine Bildschirme zum Einsatz.

Das fertige Dialog-Modell für ein Standard-Smart-Phone mit kleinem Bildschirm wird in Abbildung 106 gezeigt.

5.6. Ableitung der User-Interface-Modelle

Nachdem die Spezifikation des Domänen-Modells, der im Rahmen der Fallstudie relevanten Kontext-Modelle und der Dialog-Modelle für diese Kontexte abgeschlossen ist, kann nun mit der Konstruktion bzw. Transformation der UI-Modelle auf den verschiedenen Abstraktionsebenen begonnen werden.

5.6.1. Abstraktes User-Interface-Modell (AUI)

Das abstrakte User-Interface ist per definitionem unabhängig vom jeweiligen Benutzungskontext, d. h. es handelt sich für alle Benutzer, Endgeräte, Toolkits und Umgebungsbedingungen um dasselbe AUI.

Bei PaMGIS basiert das AUI-Modell direkt auf dem Konzept-Modell. Die in diesem enthaltenen Konzepte werden in Tabelle 35 aufgelistet.

Tabelle 35 VOLLSTÄNDIGES KONZEPT-MODELL FÜR DIE BENUTZERSCHNITTSTELLE

Konzept-ID	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
__CPT_0000_00_0001	Begrüßung	DataOutput	Begrüßung anzeigen
__CPT_0000_00_0002	Ticketkauf beenden	Activator	Reiseticket-Kauf beenden
__CPT_0001_01_0001	Ticketkauf starten	Navigator	Reiseticket kaufen
__CPT_0002_01_0001	Reiseverbindungen anfordern	Activator	Reiseverbindungen anfordern
__CPT_0003_01_0001	Startort	DataEdit	Startort eingeben
__CPT_0003_01_0002	Zielort	DataEdit	Zielort eingeben
__CPT_0003_01_0003	Zwischenstopp	DataEdit	Zwischenstopp eingeben
__CPT_0003_01_0004	Reiserichtung	SingleChoice	Reiserichtung spezifizieren
__CPT_0003_01_0005	Hinreise	ChoiceItem	Hinreise auswählen
__CPT_0003_01_0006	Hin- und Rückreise	ChoiceItem	Hin- und Rückreise auswählen
__CPT_0004_01_0001	Hinreisedatum	DataEdit	Hinreise-Datum eingeben
__CPT_0004_01_0002	Hinreisezeit	DataEdit	Hinreise-Zeit eingeben
__CPT_0004_01_0003	Interpretation Hinreisezeit	SingleChoice	Interpretation der Hinreise-Zeit
__CPT_0004_01_0004	Hinreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Hinreise)
__CPT_0004_01_0005	Hinreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Hinreise)
__CPT_0004_01_0006	Rückreisedatum	DataEdit	Rückreise-Datum eingeben
__CPT_0004_01_0007	Rückreisezeit	DataEdit	Rückreise-Zeit eingeben
__CPT_0004_01_0008	Interpretation Rückreisezeit	SingleChoice	Interpretation der Rückreise-Zeit

Konzept-ID	Konzept-Name	Konzept-Typ	Zugehörende Aufgabe
__CPT_0004_01_0009	Rückreise Abfahrtszeit	ChoiceItem	Früheste Abfahrtszeit auswählen (Rückreise)
__CPT_0004_01_0010	Rückreise Ankunftszeit	ChoiceItem	Späteste Ankunftszeit auswählen (Rückreise)
__CPT_0005_01_0001	Anzahl Erwachsene	DataEdit	Anzahl Erwachsene eingeben
__CPT_0005_01_0002	Anzahl Kinder	DataEdit	Anzahl Kinder eingeben
__CPT_0005_01_0003	Anzahl Fahrräder	DataEdit	Anzahl Fahrräder eingeben
__CPT_0005_01_0004	Anzahl Hunde	DataEdit	Anzahl Hunde eingeben
__CPT_0006_01_0001	Reiseklasse	SingleChoice	Reise-Klasse spezifizieren
__CPT_0006_01_0003	Reiseklasse 1	ChoiceItem	1. Klasse auswählen
__CPT_0006_01_0004	Reiseklasse 2	ChoiceItem	2. Klasse auswählen
__CPT_0006_01_0002	Verkehrsmittel	SingleChoice	Verkehrsmittel spezifizieren
__CPT_0006_01_0005	Verkehrsmittel 1	ChoiceItem	Alle auswählen
__CPT_0006_01_0006	Verkehrsmittel 2	ChoiceItem	Alle außer ICE auswählen
__CPT_0006_01_0007	Verkehrsmittel 3	ChoiceItem	Nur Nahverkehrsmittel auswählen
__CPT_0007_01_0001	Hinreiseverbindung	SingleChoice	Hinreise-Verbindung wählen
__CPT_0007_01_0002	Rückreiseverbindung	SingleChoice	Rückreise-Verbindung wählen
__CPT_0009_01_0001	Zahlungsart	SingleChoice	Zahlungsart spezifizieren
__CPT_0009_01_0002	Zahlungsart 1	ChoiceItem	Kreditkarte auswählen
__CPT_0009_01_0003	Zahlungsart 2	ChoiceItem	EC-Karte auswählen
__CPT_0009_01_0004	Zahlungsart 3	ChoiceItem	Kundenkarte auswählen
__CPT_0009_01_0005	Zahlungsart 4	ChoiceItem	Überweisung auswählen
__CPT_0010_01_0001	Versandart	SingleChoice	Versandart spezifizieren
__CPT_0010_01_0002	Versandart 1	ChoiceItem	Ausdrucken auswählen
__CPT_0010_01_0003	Versandart 2	ChoiceItem	Email-Versand auswählen
__CPT_0010_01_0004	Versandart 3	ChoiceItem	SMS-Versand auswählen
__CPT_0011_01_0001	Ticketkauf bestätigen	Activator	Ticketkauf bestätigen
__CPT_0012_01_0001	Eingabewert 1	DataOutput	Gewählte Verbindung Hinreise anzeigen
__CPT_0012_01_0002	Eingabewert 2	DataOutput	Gewählte Verbindung Rückreise anzeigen
__CPT_0012_01_0003	Eingabewert 3	DataOutput	Gewählte Zahlungsart anzeigen
__CPT_0012_01_0004	Eingabewert 4	DataOutput	Gewählte Versandart anzeigen

Für jedes dieser Konzepte muss nun im AUI-Modell ein namens- und typgleiches AUI-Element angelegt werden.

Bei der Erzeugung der AUI-Elemente wird jeweils ein neuer eindeutiger Bezeichner vergeben und im Beschreibungselement <AUIE_ID> hinterlegt. Die Herkunft des Elements wird mithilfe des Beschreibungselements <AUIE_Origin> dokumentiert, indem in dessen Unterelementen <AUIE_ConceptID> und <AUIE_ConceptName> der eindeutige Bezeichner und der Name des zugeordneten Konzepts eingetragen werden. Die Inhalte aller anderen Beschreibungselemente werden unverändert übernommen.

Der für diese Fallstudie relevante Ausschnitt der Zuordnungstabelle für die Domänen/AUI-Transformation wird in Tabelle 36 gezeigt. In der Spezifikation der Mapping-Tabelle bleibt das Beschreibungselement <ApplicableContextsOfUse> leer. Dies ist der Indikator dafür, dass die Tabelle unabhängig von jeglichem Benutzungskontext und somit für alle definierten Kontexte eingesetzt werden kann. Bedingungen müssen für diese Transformation nicht definiert werden.

Tabelle 36 AUSSCHNITT DER MAPPINGTABELLE FÜR DIE DOM/AUI-TRANSFORMATION

AUI-Element	Randbedingung	CUI-Element
<DataOutput>		<DataOutput>
<DataEdit>		<DataEdit>
<Activator>		<Activator>
<Navigator>		<Navigator>
<SingleChoice>		<SingleChoice>
<ChoiceItem>		<ChoiceItem>

Nach der mithilfe der Zuordnungstabelle durchgeführten Transformation erhält man das fertige AUI-Modell der Benutzerschnittstelle. Ein kurzer Ausschnitt mit den wichtigsten Beschreibungsmerkmalen ist in Tabelle 37 ersichtlich.

Tabelle 37 AUSSCHNITT DES ABSTRAKTEN UI-MODELLS

AUI-Element-ID	AUIE-Name	AUIE-Typ	Herkunft (Konzept-ID)
__AUI_0000_00000001	Begrüßung	DataOutput	__CPT_0000_00_0001
__AUI_0000_00000002	Ticketkauf beenden	Activator	__CPT_0000_00_0002
__AUI_0000_0000003	Ticketkauf starten	Navigator	__CPT_0001_01_0001
__AUI_0000_0000004	Reiseverbindungen anfordern	Activator	__CPT_0002_01_0001
__AUI_0000_0000005	Startort	DataEdit	__CPT_0003_01_0001
__AUI_0000_0000006	Zielort	DataEdit	__CPT_0003_01_0002
__AUI_0000_0000007	Zwischenstopp	DataEdit	__CPT_0003_01_0003
…	…	…	…

5.6.2. Konkrete User-Interface-Modelle (CUI)

Nun kann mit der schrittweisen Überführung des AUI- in das CUI-Modell begonnen werden. Unter Verwendung des jeweiligen Dialogmodells erfolgt zunächst eine erste Anpassung des Benutzeroberflächenmodells an den zugrundeliegenden Benutzungskontext. Im Folgenden wird diese Transformation für den Kontext „Standard-Desktop-PC“ beschrieben.

Für den ersten Dialog wird am Anfang des AUI-Modells ein <Cluster>-Element eingefügt. Dem Dialog-Modell kann entnommen werden, welche Aufgaben diesem Dialog zugeordnet sind. Die korrespondierenden Konzepte werden in das <Cluster>-Element integriert. Das Ergebnis dieses Vorgangs ist in Abbildung 107 illustriert. Die Einrückung im rechten Teil der

Grafik verdeutlicht die strukturellen Abhängigkeiten. Der Begriff AUI_{mod} steht für das wie beschrieben modifizierte AUI-Modell.

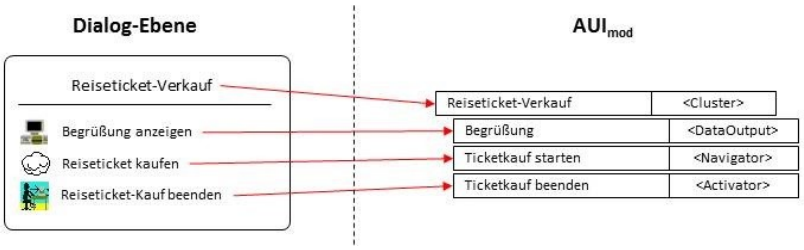


Abbildung 107 STRUKTUR DES AUI-MODELLS DES DIALOGS „REISETICKET-VERKAUF“

Mit dem Folgedialog „Reisedaten spezifizieren“ wird ebenso verfahren. Darüber hinaus enthält dieser Aufgaben, denen kein Konzept zugeordnet ist. Für solche Tasks werden als weiteres Strukturierungsmittel <Grouping>-Elemente eingefügt. In diese werden dann jeweils alle Konzepte eingefügt, die zu Aufgaben des entsprechenden Unterbaums im Aufgaben-Modell gehören. Wie in Abbildung 108 zu sehen, betrifft dies die <Grouping>-Konzepte „Reisestrecke spezifizieren“, „Reisezeit spezifizieren“, „Hinreisezeit spezifizieren“, „Rückreisezeit spezifizieren“, „Reisende spezifizieren“ und „Reiseoptionen spezifizieren“

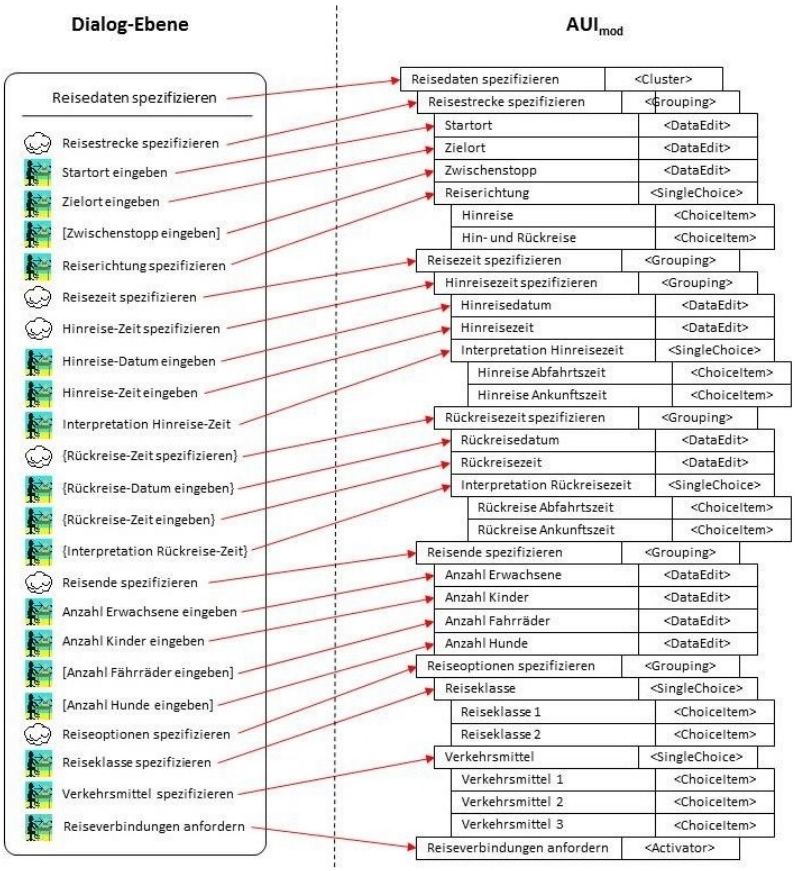


Abbildung 108 STRUKTUR DES AUI-MODELLS DES DIALOGS „REISEDATEN SPEZIFIZIEREN“

Die verbliebenen Dialoge werden analog behandelt. Es ist zu beachten, dass auch die bei der Erstellung des Dialog-Modells hinzugekommenen zusätzlichen Konzepte in den Dialogen „Reiseverbindung wählen“ und „Ausführungsart spezifizieren“ am Ende in die betreffenden

<Cluster>-Elemente aufgenommen worden sind. Die entsprechende AUI-Struktur ist in Abbildung 109 dargestellt.

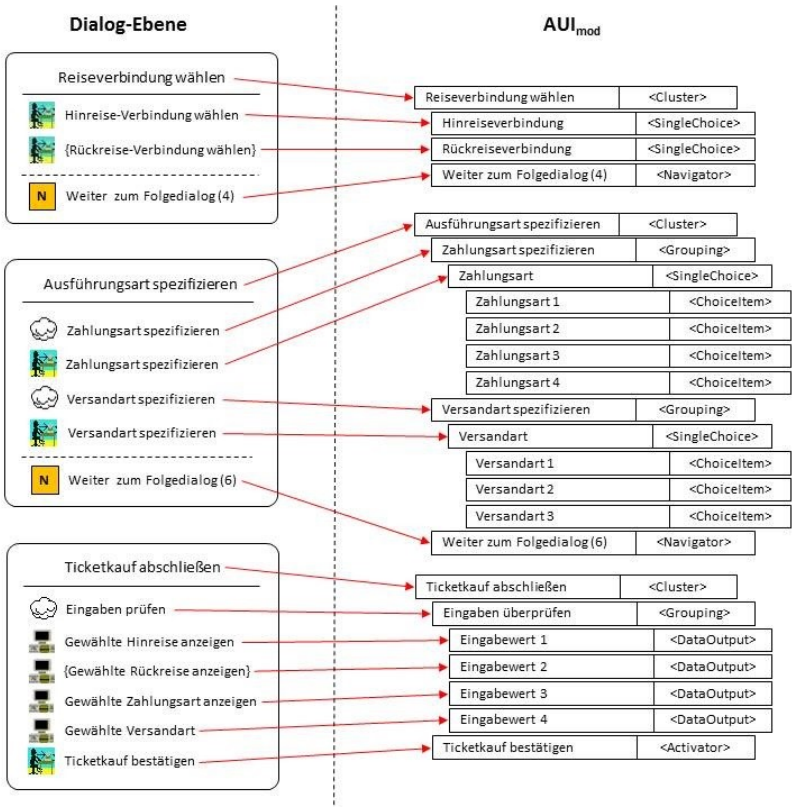


Abbildung 109 STRUKTUR DES AUI-MODELLS DER NOCH VERBLIEBENEN DIALOGE

Jetzt müssen noch die AUI-Elemente des fertig strukturierten abstrakten Modells durch CUI-Elemente ersetzt werden. Dies wird auf Basis der in Tabelle 38 gezeigten Zuordnungsliste durchgeführt.

Tabelle 38 MAPPINGTABELLE FÜR DIE AUI/CUI-TRANSFORMATION

AUI-Element	Randbedingung	CUI-Element
<Cluster>		<Form>
<Grouping>		<Box>
<DataOutput>		<OutputField>
<DataEdit>		<InputField>
<Activator>		<Button>
<Navigator>		<Button>
<SingleChoice>	Anzahl Optionen ≤ 5	<RadioButtons>
<ChoiceItem>	Anzahl Optionen ≤ 5	<RadioButton>
<SingleChoice>	Anzahl Optionen ≤ 5	<DropdownList>
<ChoiceItem>	Anzahl Optionen ≤ 5	<DropdownListEntry>
<SingleChoice>	Anzahl Optionen > 5 oder unbekannt	<ListBox>

Die sich durch die entsprechende Ersetzung ergebende Struktur des CUI-Modells für den Dialog „Reiseticket-Verkauf“ kann auf der rechten Seite von Abbildung 110 eingesehen werden.

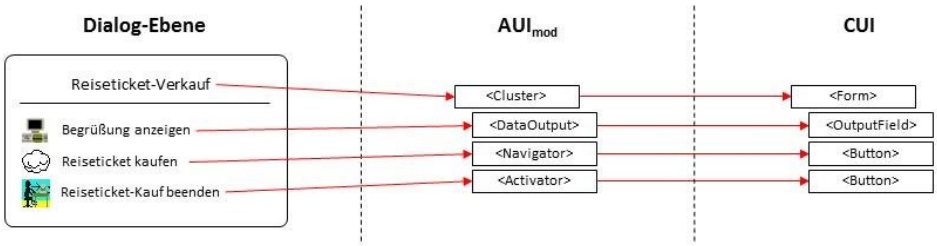


Abbildung 110 STRUKTUR DES CUI-MODELLS DES DIALOGS „REISETICKET-VERKAUF“

In gleicher Weise erfolgt die Transformation des nachfolgenden Dialogs “Reisedaten spezifizieren“.

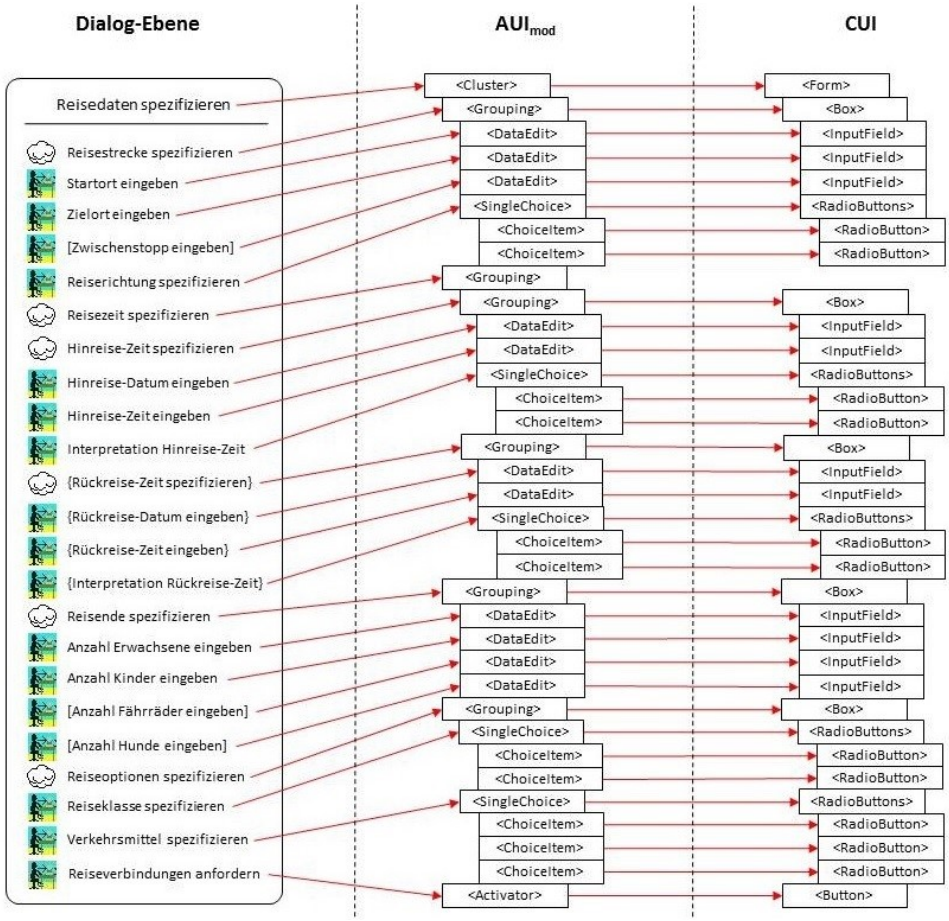


Abbildung 111 STRUKTUR DES CUI-MODELLS DES DIALOGS „REISEDATEN SPEZIFIZIEREN“

Tritt ein AUI-Element mehrfach in der linken Spalte der Zuordnungstabelle auf, so gibt es für die Ersetzung mehrere Möglichkeiten. Im vorliegenden Fall trifft dies beispielsweise auf das AUI-Element <SingleChoice> zu. Es wird dann zunächst die angegebene Randbedingung ausgewertet. Falls mehr als fünf Auswahlmöglichkeiten vorliegen oder die Anzahl der Optionen a priori nicht bekannt ist und sich erst zur Laufzeit ergibt, kommt auf CUI-Ebene eine

Listbox zum Einsatz. Existieren nur fünf oder weniger Wahlmöglichkeiten, dann erfolgt die Ersetzung entweder durch Radiobuttons oder eine Dropdownliste. Bei solchen Mehrdeutigkeiten bietet PaMGIS zwei mögliche Verfahren an:

4. Die Ersetzung erfolgt automatisch gemäß der in der Liste am weitesten oben stehende Mapping-Regel. Die Auswahl des CUI-Elements erfolgt also auf Basis der Reihenfolge der Regeln. Soll die Ersetzung geändert werden, so muss die betreffende Regel zuerst aufgeführt sein.
5. Der automatisierte Ersetzungsvorgang wird unterbrochen und die Festlegung erfolgt interaktiv durch den PaMGIS-Benutzer, dem ein entsprechender Auswahldialog angeboten wird.

Wie in Abbildung 111 dargestellt, wurden die in Dialog „Reisedaten spezifizieren“ enthaltenen <SingleChoice>-Elemente der AUI- durch Radiobuttons auf der CUI-Ebene ersetzt.

Es ist zudem anzumerken, dass der PaMGIS-User stets die Möglichkeit hat, die weitgehend automatisch ablaufende Modell-Transformation manuell zu beeinflussen bzw. zu ändern. Im vorliegenden Fall ergibt sich durch die beiden aufeinander folgenden Tasks „Reisezeit spezifizieren“ und „Hinreise-Zeit spezifizieren“ eine Verschachtelung von <Grouping>-Elementen im AUI-Modell. Dies würde letztlich zu einer inkonsistenten Ansicht in der finalen Benutzeroberfläche führen. Daher wurde das Mapping des ersten <Grouping>-Elements händisch entfernt.

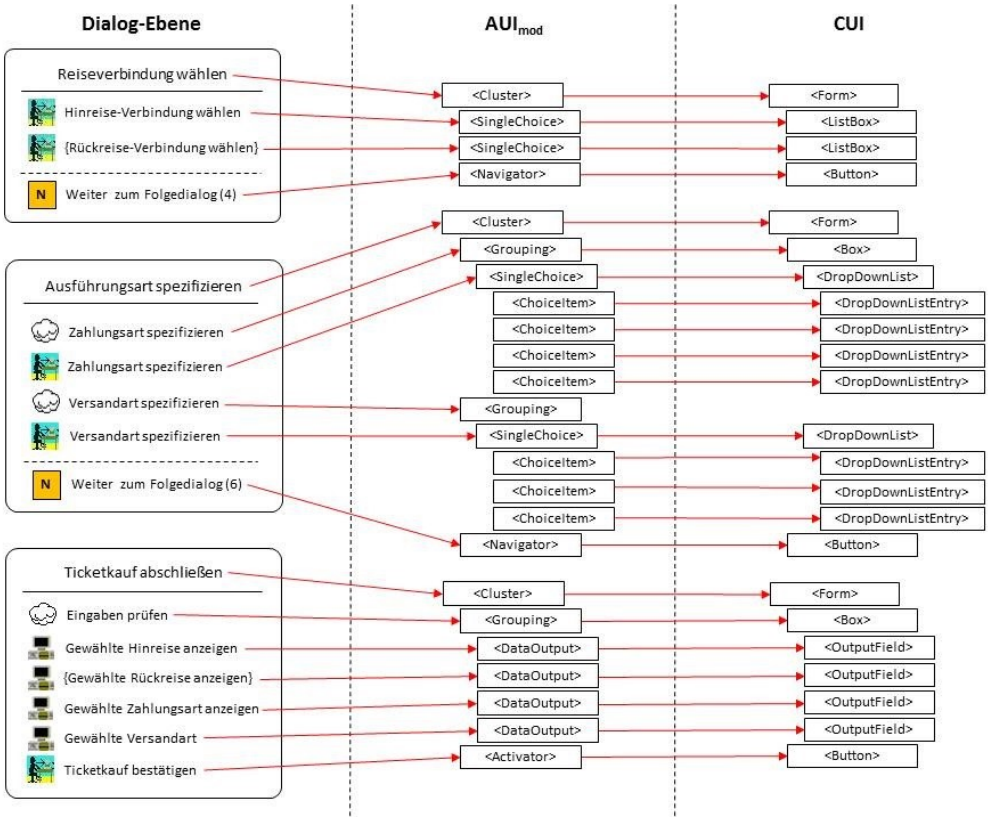


Abbildung 112 STRUKTUR DES CUI-MODELLS DER NOCH VERBLIEBENEN DIALOGE

Die verblieben Dialoge werden gleichermaßen transformiert. Es bleibt zu erwähnen, dass die <SingleChoice>-Elemente im Dialog „Ausführungsart spezifizieren“, wie in Abbildung 112 dargestellt, auf CUI-Ebene durch Dropdownlisten repräsentiert werden.

Die bisherigen Ausführungen beziehen sich auf den Benutzungskontext für große Bildschirme. Die Umwandlung des AUI- in ein CUI-Modell für den anderen in der Fallstudie betrachteten Kontext für Smartphones mit kleinem Display erfolgt in analoger Weise. Grundsätzlich können für die verschiedenen Kontexte unterschiedliche Mapping-Tabellen für die AUI/CUI-Transformation eingesetzt werden. Im vorliegenden Beispiel wurden aber dieselben Zuordnungsregeln verwendet.

5.6.3. Finale User-Interface-Modelle (FUI)

Schließlich wird aus dem CUI-Modell die finale Benutzeroberfläche generiert. Im Rahmen der Fallstudie erfolgt die prototypische Implementierung mit der Programmiersprache *Python* und dem hierfür angepassten UI-Toolkit *Tkinter*.

Wie schon bei der Überführung des AUI- in das CUI-Modell findet auch hier eine Zuordnungstabelle Verwendung. Durch sie wird festgelegt, welche CUI-Elemente durch welche Python-Objekte realisiert werden sollen. Die grundlegenden Mapping-Regeln sind in Tabelle 39 aufgelistet.

Tabelle 39 MAPPINGTABELLE FÜR DIE AUI/FUI-TRANSFORMATION

CUI-Element	FUI-Element
<Form>	Toplevel()
<Box>	Frame()
<OutputField>	Label()
<InputField>	Text()
<Button>	Button()
<RadioButtons>, <RadioButton>	Radiobutton()
<DropdownList>, <DropdownListEntry>	OptionMenu()
<Listbox>	Listbox()

In Abbildung 113 wird auf der rechten Seite die Struktur des resultierenden FUI-Modells illustriert.

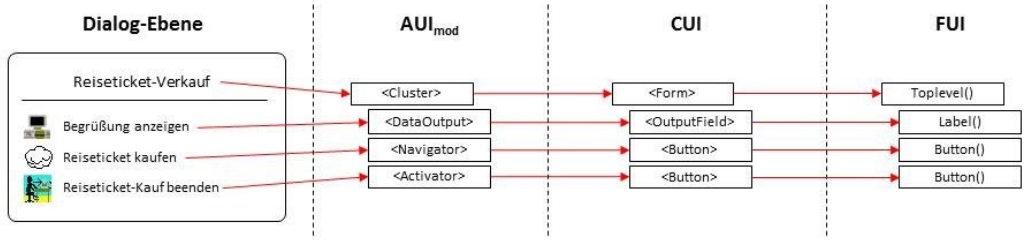


Abbildung 113 STRUKTUR DES FUI-MODELLS DES DIALOGS „REISETICKET-VERKAUF“

In einigen Fällen kann keine direkte Ersetzung von CUI-Elementen in FUI-Objekte erfolgen, sondern es sind komplexere Umformungen notwendig. Beispielsweise werden, wie dies in

Abbildung 114 skizziert ist, im Dialog „Reisedaten spezifizieren“ die <RadioButton>-Elemente der CUI-Ebene durch Tkinter-Radiobuttons realisiert. Es fließen hierbei zusätzlich Informationen aus den übergeordneten CUI-Elementen des Typs <RadioButtons> ein, die darüber hinaus aber über keine korrespondierenden FUI-Objekte verfügen. Die korrekte Transformation wird durch den UI-Generator für Python/Tkinter sichergestellt.

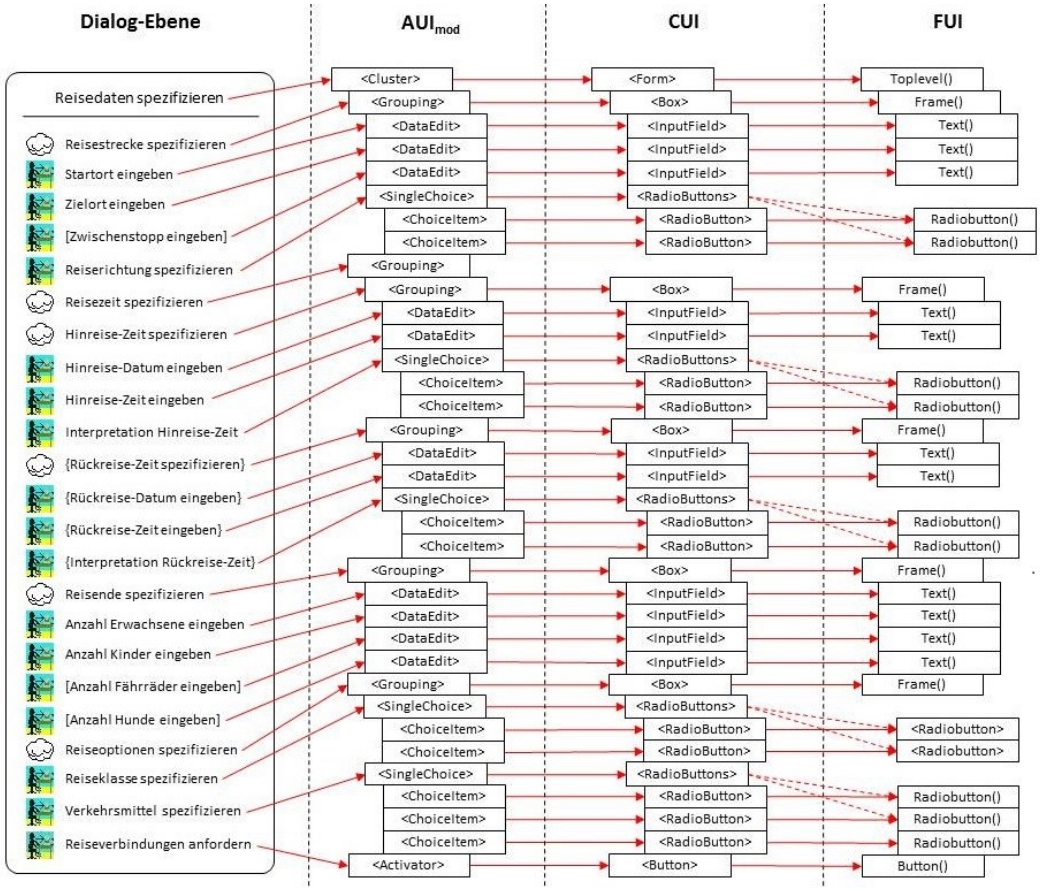


Abbildung 114 STRUKTUR DES FUI-MODELLS DES DIALOGS „REISEDATEN SPEZIFIZIEREN“

Ähnlich wie bei den Radiobuttons ist auch die Erstellung die Konstruktion eines Tkinter-OptionMenu eine komplexere Transformation. Es müssen dabei sowohl Informationen der CUI-Elemente des Typs <DropdownList> als auch der zugehörenden <DropdownListEntry>-Elemente berücksichtigt werden. Dies wird in Abbildung 115 illustriert.

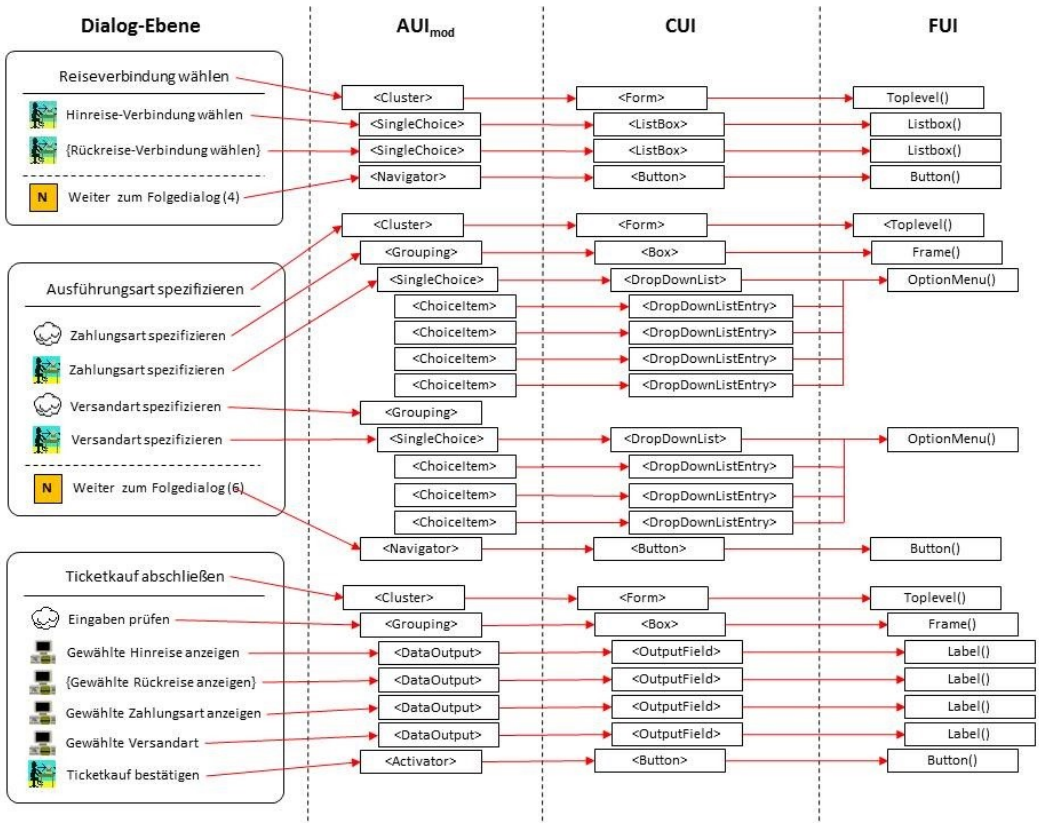


Abbildung 115 STRUKTUR DES FUI- MODELLS DER NOCH VERBLIEBENEN DIALOGE

Abschließend werden in Abbildung 116 und Abbildung 117 Screenshots von prototypisch erstellten Python/Tkinter-Dialogen für den Kontext „Standard-Desktop-PC“ gezeigt.

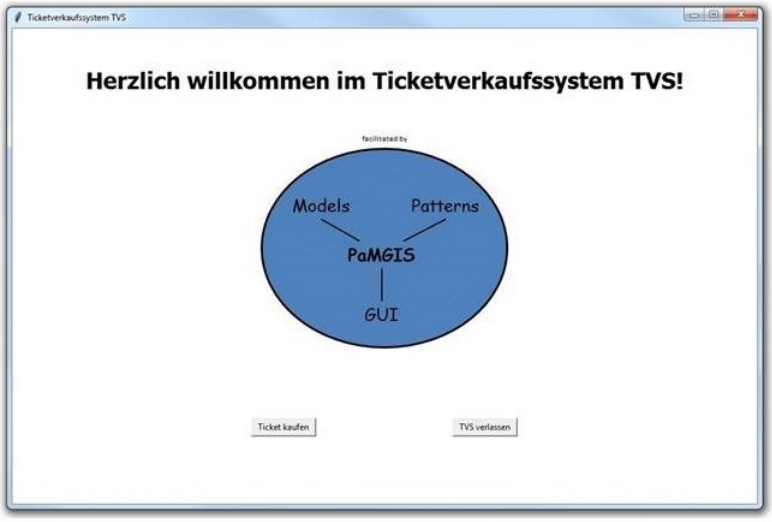


Abbildung 116 SCREENSHOT DES DIALOGS „REISETICKET-VERKAUF“ FÜR GROßE BILDSCHIRME

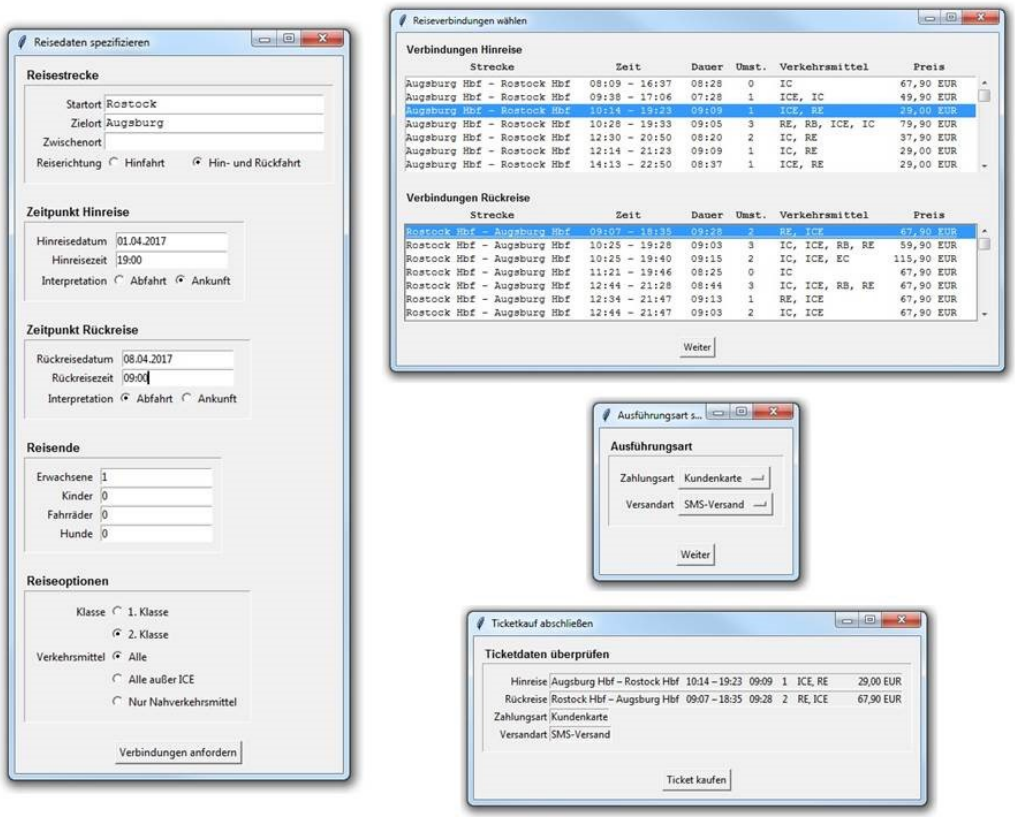


Abbildung 117 SCREENSHOTS DER WEITEREN DIALOGE FÜR GROßE BILDSCHIRME

Die Screenshots der prototypischen Implementierung für den Benutzungskontext „Standard-Smart-Phone“ mit kleinem Display halten Abbildung 118 und Abbildung 119 bereit.

Abbildung 118 SCREENSHOT DES DIALOGS „REISETICKET-VERKAUF“ FÜR KLEINE BILDSCHIRME

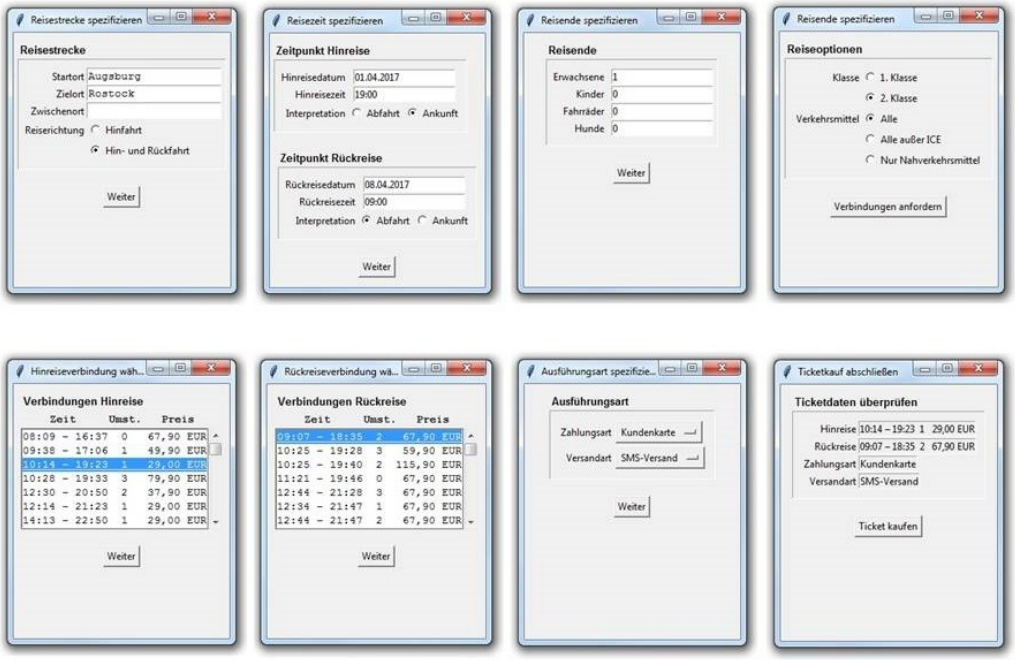


Abbildung 119 SCREENSHOTS DER WEITEREN DIALOGE FÜR KLEINE BILDSCHIRME

6. Epilog

6.1. Zusammenfassung

Diese Dissertation widmet sich dem Entwurf einer Entwicklungsumgebung für interaktive Benutzerschnittstellen, bei der einerseits ein modellgetriebener und andererseits ein musterbasierter Ansatz zu einem integrierten Entwicklungsparadigma kombiniert werden.

Ursprünglich wurde davon ausgegangen, dass nach der Analyse von bereits existierenden modellbasierten Frameworks zur UI-Entwicklung ein geeignetes ausgewählt und im Rahmen der Arbeit um einen musterbasierten Anteil erweitert wird. Dies ließ sich in dieser Form aber nicht realisieren, da zu keiner der untersuchten Entwicklungsumgebungen vollständige beziehungsweise ausreichend detaillierte Informationen zu ermitteln waren. Deshalb wurde der Umfang der Dissertation dahingehend erweitert, dass nicht nur der musterbasierte, sondern auch der modellgetriebene Anteil des neuen Frameworks gänzlich entworfen wurde.

Folglich wurden zunächst die Aspekte der modellgetriebenen Benutzeroberflächen recherchiert und analysiert. Neben den grundlegenden Verfahren und Techniken wurden insgesamt acht Modellierungssprachen und 18 bestehende Entwicklungsumgebungen analysiert und bewertet. Auch hinsichtlich der musterbasierten UI-Entwicklung wurden die fundamentalen Aspekte beleuchtet und zusammen sechs existierende formale Sprachen zur Beschreibung von Mustern, fünf weit verbreitete und allgemein anerkannte Pattern-Sammlungen sowie 14 Werkzeuge zur Verwaltung und Verarbeitung von Mustern im Detail untersucht.

Auf Basis der dadurch gewonnen Ergebnisse und Erkenntnisse erfolgte daraufhin die Festlegung der generellen Struktur und Architektur der kombinierten modellgetriebenen und musterbasierten Entwicklungsumgebung. Anschließend wurde dann der Entwurf des PaMGIS-Frameworks schrittweise verfeinert.

Um die Funktionsweise der zuvor theoretisch hergeleiteten Verfahren und Techniken am praktischen Beispiel zu erläutern und einen ersten Nachweis der Eignung des neuen Frameworks hinsichtlich der UI-Entwicklung zu erbringen, wurde eine Fallstudie erstellt. Sie beschäftigt sich mit Benutzeroberflächen für die Geschäftsdomäne des Ticketverkaufs für die Personenbeförderung; im konkreten Fall für den Bahnverkehr.

6.2. Stellungnahme zu den Forschungsfragen

In Kapitel 1.3 wurden einleitend acht Forschungsfragen aufgestellt, die sowohl zur thematischen Einordnung als auch als „Wegweiser“ für die inhaltliche Ausrichtung der Arbeit dienen. Die detaillierte Bearbeitung der betreffenden Themen kann den vorstehenden Kapiteln entnommen werden. An dieser Stelle erfolgt nun die abschließende und zusammenfassende Stellungnahme zu den genannten Forschungsfragen.

Die ersten vier Fragen beziehen sich hierbei auf den modellgetriebenen Teil des Framework-Entwurfs.

1. *Kann eine Wiederverwendung von Design-Know-how durch den Einsatz von HCI-Patterns bei der modellgetriebenen Entwicklung von interaktiven Benutzeroberflächen realisiert werden?*

Diese Frage ist eindeutig positiv zu beantworten. Bei dem kombinierten Entwicklungsansatz werden die Patterns unter anderem mit Modell-Fragmenten angereichert. Wie in Kapitel 4 beschrieben, können diese einerseits bei der Konstruktion von Domänen-Modellen, also bei der Erstellung von Aufgaben- und Konzept-Modellen, quasi wie Bausteine verwendet werden. Andererseits können andere Typen von Modell-Fragmenten, wie beispielsweise Dialog-Modell-Fragmente, bei der Ableitung beziehungsweise der Transformation der UI-Modelle der unterschiedlichen Abstraktionsebenen zum Einsatz kommen. Welche praktische Unterstützung die Verwendung von Mustern bei der modellgetriebenen Entwicklung von Benutzeroberflächen leistet, wird zudem in der Fallstudie in Kapitel 5 aufgezeigt.

2. *Wie kann eine modellgetriebene Entwicklungsumgebung für Benutzeroberflächen aufgebaut sein, damit sowohl eine - zumindest teilweise - automatisierte UI-Generierung erfolgen kann als auch ein sinnvoller Einsatz von HCI-Patterns ermöglicht wird?*

Zur Beantwortung dieser Frage wurden zunächst die Referenzmodelle *Model Driven Architecture* (MDA, siehe Kapitel 2.1.2), *Cameleon Reference Framework* (CRF, siehe Kapitel 2.1.3) und *Arch*-Modell (siehe Kapitel 2.1.4) studiert. Zudem erfolgte die Analyse und Bewertung von acht existierenden Modellierungssprachen (siehe Kapitel 2.4) und 18 modellgetriebenen UI-Entwicklungsumgebungen (siehe Kapitel 2.5). Der generelle Aufbau des PaMGIS-Frameworks, der aus den betreffenden Ergebnissen abgeleitet wurde, ist in Kapitel 4.2 beschrieben. Details zum modellgetriebenen Anteil der Entwicklungsumgebung können Kapitel 4.6 entnommen werden.

3. *Welche Modelle sollen hierbei zum Einsatz kommen und in welchen Beziehungen stehen diese zueinander?*

Die Modell-Architektur von PaMGIS hält sich im Wesentlichen an die Vorgaben des CRF. Auf der obersten Abstraktionsebene befindet sich das Domänen-Modell, das aus zwei Teilmodellen, dem Aufgaben- und dem Konzept-Modell, besteht. Auf der nächsten konkreteren Ebene befinden sich das Benutzungskontext- und das Dialog-Modell. In Abweichung zum CRF besteht ersteres aber aus vier Teilmodellen, dem Benutzer-, Geräte-, UI-Toolkit- und dem Umgebungs-Modell. Die eigentlichen User-Interface-Modelle liegen in Form eines abstrakten (AUI), konkreten (CUI) und finalen Modells vor. Das Adaptions-Modell des CRF, das aus den Evolutions- und Transitions-Teilmodellen besteht und für die dynamische Anpassung des Erscheinungsbildes der Benutzeroberfläche zur Laufzeit verantwortlich ist, wurde im Rahmen der vorliegenden Arbeit aus Komplexitätsgründen nicht berücksichtigt. Die gewählte Architektur des Frameworks erlaubt aber eine spätere Erweiterung in diese Richtung. Details zu den verwendeten Modellen und deren Beziehungen zueinander können in den Kapiteln 4.1.1, 4.4.3 und 4.6.2 nachgelesen werden.

4. *In welcher Weise können diese Modelle formal beschrieben werden, damit eine sinnvolle Kombination mit HCI-Patterns möglich ist?*

Für die formale Spezifikation der Modelle wurde eine XML-konforme Beschreibungssprache gewählt. Einerseits ist eine solchermaßen strukturierte und textbasierte Modellbeschreibung auch ohne besondere Werkzeuge durch die PaMGIS-Benutzer lesbar und kann auch ohne tiefer gehendes Softwareentwicklungs-Know-how verstanden werden. Andererseits

besteht so gegebenenfalls auch die Möglichkeit, anderweitig definierte Modelle oder Modellteile, wie beispielsweise UML-Klassenmodelle, mit bereits vorhandenen Transformationsmechanismen in das PaMGIS-Format zu übersetzen. Detaillierte Angaben zu den PaMGIS-Modellbeschreibungen befinden sich in Kapitel 4.6.2 und Anhang E.

Die fünfte und sechste Forschungsfrage sind dem musterbasierten Anteil der entworfenen Entwicklungsumgebung gewidmet.

5. *Wie können Muster derart formalisiert beschrieben werden, dass sie einerseits maschinell zu verarbeiten sind und andererseits eine modellgetriebene Generierung von Benutzeroberflächen sinnvoll ergänzen?*

Zur Beantwortung dieser Frage wurden zunächst sechs existierende Musterbeschreibungssprachen (siehe Kapitel 3.2.1) und fünf populäre Pattern-Sammlungen (siehe Kapitel 3.3.2) analysiert und bewertet. Auf Basis der hierdurch gewonnen Erkenntnisse wurde die *PaMGIS Pattern Specification Language* (PPSL) entwickelt, die in zweierlei Hinsicht als *Lingua franca* angesehen werden kann. Auf der einen Seite ist sie so aufgebaut, dass eine Überführung von bereits bestehenden Mustern aus den untersuchten Pattern-Sammlungen in das PPSL-Format mit möglichst geringem Informationsverlust erfolgen kann. Auf der anderen Seite werden die Pattern-Spezifikationen um weitere Informationen, wie beispielsweise die Modell-Fragmente, erweitert, so dass der modellgetriebene Teil des Frameworks sinnvoll ergänzt wird. Aus den gleichen Gründen wie bei den Modell-Spezifikationen erfolgt auch die Beschreibung der Muster in XML-konformer Weise. Details zum Aufbau von PPSL können Kapitel 4.7.2 sowie Anhang G entnommen werden. Die Bedeutung von PPSL im kombinierten modellgetriebenen und musterbasierten Entwicklungsansatz wird in Kapitel 4.4 verdeutlicht, praktische Aspekte zeigt die Fallstudie in Kapitel 5 auf.

6. *Ist es möglich, bereits existierende Muster in diesen neuen Formalismus zu überführen und so zu erweitern, dass sie bei einem kombinierten Entwicklungsansatz wiederverwendet werden können?*

Wie bereits oben erwähnt, war eines bei der Entwicklung von PPSL verfolgten Ziele eine größtmögliche Kompatibilität zu den untersuchten Beschreibungssprachen und Pattern-Sammlungen. Im Rahmen der Analyse und Bewertung der untersuchten Pattern-Sammlungen (siehe Kapitel 3.3) wurde der Nachweis erbracht, dass alle in diesen Bibliotheken enthalten Muster ohne Informationsverlust in das Format der Beschreibungssprache PLML 1.1 übertragbar sind. Die Details hierzu befinden sich in Anhang C.6. Es bleibt aber festzuhalten, dass wegen der Existenz von mehrdeutigen Zuordnungsbeziehungen zwischen den jeweiligen Beschreibungsattributen eine vollständige Automatisierung der Transformation in das PLML 1.1-Format nicht möglich ist. Zusätzlich wurde noch der Beweis erbracht, dass sich die untersuchten Musterbeschreibungssprachen (siehe Kapitel 3.2.1), insbesondere auch PLML 1.1, in das PPSL-Format transformieren lassen. Die entsprechenden detaillierten Mapping-Tabellen sind in Anhang H ersichtlich. Eine Ausnahme hierzu bildet die Sprache XPLML, zu deren konkreten Aufbau in der für die Recherchearbeiten vorliegenden Literatur keine Hinweise zu finden waren.

Die letzten drei Fragen zielen auf den kombinierten modellgetriebenen und musterbasierten UI-Entwicklungsansatz.

7. *Aus welchen Schritten besteht die neue, kombinierte Entwicklungsmethodik und in welcher zeitlichen Abfolge stehen diese?*

Die integrierte modellgetriebene und musterbasierte Entwicklungsmethodik wird ausführlich in Kapitel 4.4 erläutert. Insbesondere wird die Abfolge der einzelnen Schritte des Prozesses in Abbildung 17 in Kapitel 4.4.3 aufgezeigt.

8. *Welche Hilfsmittel und Werkzeuge zur Unterstützung der Benutzer des Frameworks sind hinsichtlich der kombinierten Entwicklungsmethodik sinnvoll beziehungsweise notwendig?*

Zur Beantwortung dieser Frage wurden zunächst 14 existierende Werkzeuge zur Verwaltung und Verarbeitung von Mustern untersucht (siehe Kapitel 3.4). Zudem wurde dieses Thema auch bei der Analyse der 18 modellgetriebenen UI-Entwicklungsumgebungen (siehe Kapitel 2.5) berücksichtigt. Auf Basis der entsprechenden Ergebnisse und beim Entwurf des integrierten modellgetriebenen und musterbasierten Entwicklungsprozesses wurden insgesamt 14 Gruppen unterschiedlicher Verfahren und Funktionalitäten identifiziert, die potenziell durch Software-Werkzeuge unterstützt werden können, beziehungsweise aufgrund ihrer Komplexität in Tools abgebildet werden müssen. Es ist jedoch nicht zwingend erforderlich, für jede dieser Funktionen ein separates Werkzeug vorzusehen – die Zusammenfassung einiger Funktionalitäten in gemeinsamen Tools ist sicherlich möglich und sinnvoll. Bei den 14 identifizierten Funktionsblöcken beziehen sich jeweils drei auf die Erstellung und Verwaltung der Modelle und Patterns, die verbleibenden acht auf den integrierten Entwicklungsprozess. Einen Überblick über alle vorgesehenen Werkzeuge gibt Abbildung 38 in Kapitel 4.6.1. Die Beschreibungen der Tools sind in den Kapiteln 4.6.1 und 4.7.1 zu finden.

9. *Führt die kombinierte Nutzung von Mustern und Modellen in einer gemeinsamen Entwurfs- und Entwicklungsumgebung zu besseren Benutzerschnittstellen oder höherer Effizienz bei deren Entwicklung im Vergleich zu reinen modellgetriebenen bzw. musterbasierten Ansätzen?*

Die Frage nach der Effizienz des kombinierten Entwicklungsansatzes lässt sich zunächst nur qualitativ beantworten. Bei der Verwendung einer musterbasierten Vorgehensweise kann zwar bestimmungsgemäß durch die Patterns entsprechendes Design-Know-how wiederverwendet und somit der Entwurf von Benutzeroberflächen gegenüber der klassischen UI-Entwicklung beschleunigt werden. Eine Anpassung an verschiedene Benutzungskontexte, also beispielsweise an signifikant unterschiedliche Benutzer, Endgeräte und/oder Software-Plattformen muss hierbei jedoch durch eine, zumindest teilweise, Neuentwicklung der Benutzerschnittstellen erfolgen. Gegebenenfalls kommen für verschiedene Kontexte auch unterschiedliche Muster, die gegebenenfalls auch aus unterschiedlichen Pattern-Sammlungen herausgesucht werden müssen, zum Einsatz. Aufgrund der in den Kapiteln 3.2 und 3.3 festgestellten, inkonsistenten Beschreibung der Muster und der fehlenden sammlungsübergreifenden Definition von Beziehungen zwischen den Patterns, muss jeder einzelne Entwickler immer wieder erneut viel Arbeit hierfür investieren. Bei der Verwendung des PaMGIS-Frameworks fällt natürlich Aufwand an, um die Muster entweder neu zu spezifizieren oder bereits existierende Patterns in das PPSL-Format zu übersetzen und die fehlenden Relationen zu definieren. Im Gegensatz zum rein musterbasierten Vorgehen fällt dieser Aufwand jedoch nur einmalig an.

Durch rein modellgetriebene Ansätze wird die Anpassung an verschiedene Benutzungskontexte, wie in den Kapitel 2.1.2 und 2.1.3 erläutert, unterstützt und vereinfacht. Es werden diverse Modelle erstellt und die Benutzeroberfläche kann dann jeweils schrittweise, über die verschiedenen Abstraktionsebenen hinweg, zumindest teilweise automatisiert

generiert werden. Das Aufbauen der Modelle nimmt jedoch vergleichsweise viel Zeit in Anspruch. Durch die Verwendung von Patterns kann dieser Aufwand durchaus gesenkt werden. Die in den Mustern enthaltenen Modell-Fragmente können praktisch wie vorgefertigte Bausteine verwendet werden und somit die Modellerstellung beschleunigen. Zudem kann somit auch das Fehlerpotenzial bei der Konstruktion der Modelle gesenkt werden. Aufgrund der genannten Vorteile des kombinierten Ansatzes gegenüber der reinen musterbasierten bzw. modellgetriebenen UI-Entwicklung besteht die berechtigte Hoffnung, dass Benutzeroberflächen mit PaMGIS effizienter entwickelt werden können. Endgültigen Aufschluss können letztlich nur Vergleichsstudien, wie sie in Kapitel 6.3 vorgeschlagen werden, geben.

Ähnlich verhält es sich auch bezüglich der Qualität der entwickelten Benutzerschnittstellen. Im Unterschied zur reinen modellgetriebenen UI-Entwicklung besitzt der integrierte PaMGIS-Ansatz den Vorteil, dass der Framework-Benutzer auf einfache Art und Weise interaktiv durch die Anwendung unterschiedlicher Patterns die Modelle und somit auch die resultierende Benutzeroberfläche verändern kann. Zudem lassen sich einmal durchgeführte Änderungen auch schnell wieder rückgängig machen. Dies ist bei einem ausschließlich modellgetriebenen Verfahren kaum und nur mit vergleichsweise hohem Aufwand möglich. Wie schon bei der Effizienz, lässt sich ein endgültiges Urteil hinsichtlich der Qualität der Ergebnisse nur auf Basis von Vergleichsstudien fällen.

6.3. Ausblick

Ausgehend von dem im Rahmen dieser Dissertation erarbeiteten und beschriebenen Entwurfs der integrierten modellgetriebenen und musterbasierten UI-Entwicklungsumgebung PaMGIS sind folgende Möglichkeiten zur Weiterentwicklung und Forschung vorstellbar:

1. Durchführung von Entwicklungsprojekten mit PaMGIS
   Mithilfe des integrierten modellgetriebenen und musterbasierten Verfahrens können entsprechende Projekte zur Modellierung und Generierung von Benutzeroberflächen durchgeführt werden.

2. Verbesserung der Usability von PaMGIS
   Die Gebrauchstauglichkeit des PaMGIS-Frameworks kann durch Feedback aus den Entwicklungsprojekten (siehe Punkt 1) und die Durchführung von Usability-Tests evaluiert und gegebenenfalls kontinuierlich verbessert werden.

3. Durchführung von Untersuchungen zur Ermittlung der Effizienz von PaMGIS
   Sehr interessant wäre die Durchführung von Studien zur Effektivität und Effizienz der PaMGIS-Entwicklungsumgebung. Durch geeignete Vergleiche von traditioneller und integrierter modellgetriebener und musterbasierter UI-Entwicklung könnten Aufschlüsse bezüglich des benötigten Implementierungsaufwands und auch der Qualität der resultierenden Benutzeroberflächen ermittelt werden.

4. Verbesserung des Modell-Konzepts des PaMGIS-Frameworks
   Das Framework könnte um weitere Modelle, beispielsweise um Präsentations- und/oder Layout-Modelle erweitert werden, um die Anzahl der manuellen Eingriffe bei der UI-Generierung zu reduzieren und die Qualität der Benutzeroberflächen zu

erhöhen. Zudem würde ein Werkzeug zur Validierung der Modelle dazu beitragen, Inkonsistenzen zu vermeiden und dadurch die Fehleranfälligkeit zu verringern.

5. Höherer Automatisierungsgrad durch bessere Ausnutzung der Modelle
   Es kann untersucht werden, ob und wie die in den Modellen vorhandenen Informationen dazu genutzt werden können, den Automatisierungsgrad hinsichtlich der Erstellung und Transformation der Modelle zu erhöhen. Beispielsweise können Inhalte des Benutzungskontext-Modells, insbesondere des User- und des Geräte-Modells, in die Kontruktion des Dialog-Modells einfliessen.

6. Erweiterung des PaMGIS-Frameworks um die *Usability-Feedback*-Komponente
   Detaillierter Entwurf und Implementierung der ursprünglich geplanten *Usabiliy Feedback*-Komponente, die aufgrund der Scope-Änderung der Dissertation nicht näher betrachtet wurde (siehe Kapitel 4).

7. Erweiterung des PaMGIS-Frameworks zur Generierung adaptiver User-Interfaces
   Sehr interessant ist auch der Aspekt der Anpassung der Benutzeroberfläche an den jeweils aktuellen Benutzungskontext zur Laufzeit. Hierzu müssten Veränderungen am Kontext identifiziert und geeignet darauf reagiert werden. Zu diesem Zweck könnte das Modell-Konzept von PaMGIS um die Adaptions-Modelle des CRF, d.h. um die Evolutions- und Transitions-Teilmodelle, erweitert werden. Zudem würde eine Laufzeitumgebung zur Interpretation der UI-Modelle und Darstellung des User-Interface notwendig werden. Die im Rahmen des CRF beschriebenen Verfahren zur Anpassung der Benutzeroberfläche könnten direkt übernommen werden. Es wäre auch zu untersuchen, welche Arten von Kontextänderungen auf welche Abstraktionsebenen der UI-Modelle (AUI, CUI, FUI) Einfluss haben und wie die Änderung der Benutzeroberfläche erfolgen muss, damit der Benutzer nicht verwirrt und die Usability erhalten bleibt. Besondere Aufmerksamkeit sollte hierbei wieder der Verwendung von Patterns geschenkt werden. Naheliegend ist die Vermutung, dass nicht nur das Erscheinungsbild des User-Interface durch HCI-Patterns beeinflusst werden kann, sondern auch die UI-Transformationen bei Kontextwechseln durch Patterns beschreibbar sind.

8. Reverse-Engineering der User-Interface-Modelle
   Hierbei handelt es sich praktisch die Umkehrung des Generierungsprozesses. Zu untersuchen wären die zur Verfügung stehenden Möglichkeiten, von konkreteren auf die abstrakten UI-Modelle zu schließen. So wäre es beispielsweise möglich, aus einem vorhandenen FUI durch Reverse-Engineering schrittweise das CUI-, AUI- und gegebenenfalls sogar das Domänen-Modell zu erhalten. Somit könnten im Idealfall aus existierenden, traditionell entwickelten Benutzeroberflächen die betreffenden, für die Verwendung mit PaMGIS benötigten, Modelle erzeugt werden. Zudem würde Reverse-Engineering der UI-Modelle einen wertvollen Beitrag zur UI-Adaption zur Laufzeit (siehe Punkt 7) leisten.

7. Literaturverzeichnis

[1] B. A. Myers, „A Brief History of Human-computer Interaction Technology," *interactions, Volume 5, Issue 2,* pp. 44-54, March/April 1998.

[2] A. R. Hevner, S. T. March, J. Park und S. Ram, „Design Science in Information Systems Research," *MIS Quarterly, Volume 28, Issue 1,* pp. 75-105, March 2004.

[3] J. Miller und J. Mukerji (eds.), „MDA Guide Version 1.0.1," 12th June 2003. [Online]. Available: http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf. [Zugriff am 14 06 2014].

[4] K. Czarnecki und S. Helsen, „Feature-based Survey of Model Transformation Approaches," *IBM Systems Journal - Model-driven Software Development, Volume 45, Issue 3,* pp. 621-645, July 2006.

[5] F. Truyen, „The Fast Guide to Model Driven Architecture - The Basics of Model Driven Architecture," Cephas Consulting Corp., January 2010. [Online]. Available: http://www.omg.org/mda/mda_files/Cephas_MDA_Fast_Guide.pdf. [Zugriff am 14 06 2014].

[6] T. Mens und P. Van Gorp, „A Taxonomy of Model Transformation," *Electronic Notes in Theoretical Computer Science (ENTCS), Volume 152,* pp. 125-142, March 2006.

[7] „OMG Homepage," Object Management Group, [Online]. Available: http://www.omg.org/. [Zugriff am 14 06 2014].

[8] M. Kempa und Z. Á. Mann, „Model Driven Architecture," *Informatik Spektrum,* pp. 298-302, 01 08 2005.

[9] S. J. Mellor, K. Scott, A. Uhl und D. Weise, „Model-Driven Architecture," in *OOIS 2002 Workshops, LNCS 2426, pp. 290-297*, Montpellier, France, 2002.

[10] „CAMELEON Project Homepage," CAMELEON Project, [Online]. Available: http://giove.isti.cnr.it/projects/cameleon.html. [Zugriff am 21 06 2014].

[11] „CAMELEON Project Flyer," [Online]. Available: http://giove.isti.cnr.it/projects/cameleon/pdf/flyer.pdf. [Zugriff am 21 06 2014].

[12] G. Calvary, J. Coutaz, L. Bouillon, M. Florins, Q. Limbourg, L. Marucci, F. Paternò, C.

Santoro, N. Souchon, D. Thevenin und J. Vanderdonckt, „The CAMELEON Reference Framework," 2002. [Online]. Available: http://giove.isti.cnr.it/projects/cameleon/pdf/CAMELEON%20D1.1RefFramework.pdf. [Zugriff am 09 09 2013].

[13] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon und J. Vanderdonckt, „A unifying reference framework for multi-target user interfaces," *INTERACTING WITH COMPUTERS Vol. 15,* pp. 289-308, 2003.

[14] L. Balme, A. Demeure, N. Barralon, J. Coutaz und G. Calvary, „CAMELEON-RT: A Software Architecture Reference Model for Distributed, Migratable, and Plastic User Interfaces," in *Proceedings of Second European Symposium, EUSAI 2004, pp. 291-302*, Eindhoven, Niederlande, November 8-11, 2004.

[15] G. Calvary, J. Coutaz, D. Thevenin, L. Bouillon, M. Florins, Q. Limbourg, N. Souchon, J. Vanderdonckt, L. Marucci, F. Paternò und C. Santoro, „The CAMELEON Glossary," 13 10 2003. [Online]. Available: http://giove.isti.cnr.it/projects/cameleon/pdf/CAMELEON-D1.1%20Companion.pdf. [Zugriff am 21 06 2014].

[16] J. Coutaz und S. Balbo, „Applications: A Dimension Space for User Interface Management Systems," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 27-32*, New Orleans, Louisiana, USA, 1991.

[17] L. Bass, R. Faneuf, R. Little, N. Mayer, B. Pellegrino, S. Reed, R. Seacord, S. Sheppard und M. R. Szczur, „A Metamodel for the Runtime Architecture of an Interactive System," in *SIGCHI Bulletin, Vol. 24, No. 1, The UIMS Developers Workshop*, 1992.

[18] E. Schlungbaum, „Model-Based User Interface Software Tools - Current State of Declarative Models," Graphics, Visualization & Usability Center, Georgia Institute of Technology, Atlanta, USA, GVU Tech Report, 1996.

[19] J. Van den Bergh und K. Coninx, „Model-based Design of Context-sensitive Interactive Applications: A Discussion of Notations," in *Proceedings of the 3rd Annual Conference on Task Models and Diagrams (TAMODIA '04), pp. 43-50*, Prague, Czech Republic, 2004.

[20] F. Radeke, P. Forbrig, A. Seffah und D. Sinnig, „PIM Tool: Support for Pattern-Driven and Model-Based UI Development," in *Proceedings of 5th International Workshop on Task Models and Diagrams for User Interface Design, TAMODIA 2006, pp. 82-96*, Hasselt, Belgium, 2007.

[21] E. Schlungbaum und T. Elwert, „Automatic User Interface Generation from Declarative Models," in *Proceedings of CADUI'96, pp. 3-18*, Namur, Belgium, 1996.

[22] T. Elwert, „Continuous and Explicit Dialogue Modelling," in *Conference Companion on Human Factors in Computing Systems CHI '96, pp. 265-266*, Vancouver, British Columbia, Canada, 1996.

[23] J. M. Vanderdonckt und F. Bodart, „Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection," in *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, pp. 424-429*, Amsterdam, The Netherlands, 1993.

[24] F. Bodart, A.-M. Hennebert, J.-M. Leheureux, I. Sacré und J. Vanderdonckt, „Architecture Elements for Highly-Interactive Business-Oriented Applications," in *Selected Papers from the Third International Conference on Human-Computer Interaction (EWHCI '93), pp. 83-104*, Springer-Verlag London, UK, 1993.

[25] A. G. Kleppe, J. Warmer und W. Bast, MDA Explained: The Model Driven Architecture: Practice and Promise, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

[26] K. Czarnecki und S. Helsen, „Classification of Model Transformation Approaches," in *OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.

[27] N. Mitoussis und D. Macos, „MDA Transformation Languages - Exploration and Comparison of Capability and Quality Characteristics," in *Proceedings of 3rd Workshop of the Special Interest Group „Model-Driven Software Engineering" (MDSE08)*, Berlin, Germany, 2008.

[28] P. Huber, The Model Transformation Jungle - An Evaluation and Extension of Existing Approaches, Technische Universität Wien, Institut für Softwaretechnik und Interaktive Systeme, Wien, Österreich, 2008.

[29] K. Czarnecki und M. Antkiewicz, „Mapping Features to Models: A Template Approach Based on Superimposed Variants," in *Proceedings of the 4th International Conference on Generative Programming and Component Engineering (GPCE'05), pp. 422-437*, Tallinn, Estonia, 2005.

[30] F. Jouault, F. Allilaire, J. Bézivin, I. Kurtev und P. Valduriez, „ATL: A QVT-like Transformation Language," in *Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications (OOPSLA '06)*,

Portland, Oregon, USA, 2006.

[31] E. Gamma, R. Helm, R. Johnson und J. Vlissides, Design Patterns: Elements of Reusable Object-oriented Software, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

[32] A. Rentschler, „Model-To-Text Transformation Languages," in *Modellgetriebene Software-Entwicklung - Architekturen, Muster und Eclipse-basierte MDA (Interner Bericht 2006-18)*, Karlsruhe, Germany, Universität Karlsruhe – Fakultät für Informatik, 2006, pp. 98-129.

[33] R. Schaefer, „A Survey on Transformation Tools for Model Based User Interface Development," in *Proceedings of 12th International Conference HCI International 2007, Part I*, Beijing, China, 2007.

[34] I. Kurtev, „eclipse Open Model CourseWare (OMCW) Homepage," [Online]. Available: https://eclipse.org/gmt/omcw/resources/chapter10/downloads/IntroductionToMOFQVT.INRIA.ppt. [Zugriff am 20 02 2015].

[35] Object Management Group, „MOF 2.0 Query / Views / Transformations RFP," 24 04 2002. [Online]. Available: http://www.omg.org/cgi-bin/doc?ad/2002-4-10. [Zugriff am 20 02 2015].

[36] Object Management Group, „Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification - Version 1.1," January 2011. [Online]. Available: http://www.omg.org/spec/QVT/1.1. [Zugriff am 20 02 2015].

[37] M. Biehl, „Literature Study on Model Transformations," Technical Report ISRN/KTH/MMK/R-10/07-SE, Royal Institute of Technology, Stockholm, Sweden, 2010.

[38] N. Souchon und J. Vanderdonckt, „A Review of XML-compliant User Interface Description Languages," in *Interactive Systems. Design, Specification, and Verification*, J. Jorge, N. Jardim Nunes und J. Falcão e Cunha, Hrsg., Springer Berlin Heidelberg, 2003, pp. 391-401.

[39] J. Engel, C. Herdin und C. Märtin, „Review of User Interface Description Languages," in *Proceedings of 6. Forum Medientechnik - Next Generation, New Ideas*, St. Pölten, Austria, vwh, 2014.

[40] M. F. Ali, M. A. Pérez-Quiñones, M. Abrams und E. Shell, „Building Multi-Platform User Interfaces with UIML," in *Computer-Aided Design of User Interfaces III*, C. Kolski

und J. Vanderdonckt, Hrsg., Kluwer Academics Publishers, 2002, pp. 255-266.

[41] J. Helms et al., „UIML Version 4.0 Committee Draft," 23 01 2008. [Online]. Available: http://www.oasis-open.org/committees/download.php/28457/uniml-4.0-cd01.pdf. [Zugriff am 09 09 2013].

[42] Heise Online, „Schachtelsatz," 2005. [Online]. Available: http://www.heise.de/ix/artikel/Schachtelsatz-506516.html. [Zugriff am 09 09 2013].

[43] „XUL Tutorial with Examples," 2012. [Online]. Available: http://www.xul.fr/tutorial/. [Zugriff am 09 09 2013].

[44] „Mozilla Developer Network: XUL Reference," 2013. [Online]. Available: https://developer.mozilla.org/de/docs/XUL_Reference. [Zugriff am 09 09 2013].

[45] „Mozilla Developer Network: Gecko," 2013. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/Gecko. [Zugriff am 09 09 2013].

[46] A. Puerta und J. Eisenstein, „XIML: A Universal Language for User Interfaces," 2001. [Online]. Available: http://www.ximl.org/pages/Docs.asp. [Zugriff am 09 09 2013].

[47] „XIML Homepage," 2004. [Online]. Available: http://www.ximl.org/. [Zugriff am 09 09 2013].

[48] A. Puerta und J. Eisenstein, „XIML: A Common Representation for Interaction Data," in *Paper presented at the meeting of the IUI '02*, 2002.

[49] S. Crowle und L. Hole, „ISML: An Interface Specification Meta-Language," in *Proceedings of Interactive Systems International Workshop*, 2003.

[50] S. Crowle, The Design and evaluation of the specification framework for user interface design, PhD Thesis, Bournemouth University, UK, 2003.

[51] Q. Limbourg, J. Vanderdonckt, B. Michotte, L. Bouillon, M. Florins und D. Trevisan, „UsiXML: A User Interface Description Language for Context-Sensitive User Interfaces," in *Proceedings of the ACM AVI'2004 Workshop "Developing User Interfaces with XML: Advances on User Interface Description Languages"*, 2004.

[52] ITEA2, „UsiXML Consortium: State of the Art of User Interface Description Languages," 2013. [Online]. Available: http://www.itea2.org/project/workpackage-document/download?document=468&file=08026_UsiXML_WP1_D1_1_v2_State_of_the_Art_of_UIDL (1).doc. [Zugriff am 09 09 2013].

[53] UsiXML Consortium, „W3C Member Submission: User Interface Extensible Markup Language (UsiXML),“ 01 02 2012. [Online]. Available: http://www.w3.org/wiki/images/5/5d/UsiXML_submission_to_W3C.pdf. [Zugriff am 09 09 2013].

[54] UsiXML Consortium, „UsiXML v1.8 Reference Manual,“ 2007. [Online]. Available: https://smv.unige.ch/research-projects/batic3s/internal-documents/UsiXML_v1.8.0-Documentation.pdf. [Zugriff am 09 09 2013].

[55] Q. Limbourg und J. Vanderdonckt, „UsiXML: A User Interface Description Language Supporting Multiple Levels of Independence,“ in *ICWE Workshops*, 2004.

[56] H. Traetteberg, „Integrating Dialog Modeling and Domain Modeling: The Case of DiaMODL and the Eclipse Modeling Framework,“ *Journal of Universal Computer Science, vol. 14, no. 19,* pp. 3265-3278, 2008.

[57] H. Trætteberg, „Dialog Modelling with Interactors and UML Statecharts - a Hybrid Approach,“ in *Interactive Systems. Design, Specification, and Verification*, J. Jorge, N. Jardim Nunes und J. Falcão e Cunha, Hrsg., Springer Berlin Heidelberg, 2003, pp. 289-301.

[58] S. Berti, G. Mori, F. Paternò und C. Santoro, „TERESA: An Environment for Designing Multi-Device Interactive Services,“ 2005. [Online]. Available: http://giove.isti.cnr.it/attachments/publications/2005-A2-80.pdf. [Zugriff am 09 09 2013].

[59] „TERESA Homepage,“ ISTI, 2010. [Online]. Available: http://giove.isti.cnr.it/teresa.html. [Zugriff am 09 09 2013].

[60] ISTI, „MARIA Fact Sheet,“ 2011. [Online]. Available: http://giove.isti.cnr.it/tools/MARIA/MARIA%20Fact%20Sheet.pdf. [Zugriff am 09 09 2013].

[61] S. Berti, F. Correani, F. Paternò und C. Santoro, „The TERESA XML Language for the Description of Interactive Systems at Multiple Abstraction Levels,“ in *Proceedings Workshop on Developing User Interfaces with XML: Advances on User Interface description Languages*, 2004, pp. 103-110.

[62] G. Mori, F. Paternò und C. Santoro, „Design and Development of Multidevice user Interfaces through Multiple Logical Descriptions,“ *Journal IEEE Transactions on Software Engineering, Volume 30 Issue 8,* pp. 507-520, August 2004.

[63] F. Paternò, „The ConcurTaskTrees Notation,“ in *Model-Based Design and Evaluation of Interactive Applications*, Springer Berlin Heidelberg, 2000, pp. 39-66.

[64] F. Paternò, C. Santoro und L. D. Spano, „Model-based Design of Multi-device Interactive Applications Based on Web Services,“ in *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction, Part I*, 2009.

[65] F. Paternò, C. Santoro und L. D. Spano, „MARIA: A Universal, Declarative, Multiple Abstraction-Level Language for Service-Oriented Applications in Ubiquitous Environments,“ in *ACM Transactions on Human-Computer Interaction*, 2009.

[66] F. Paternò, C. Santoro und L. D. Spano, „A Design Space for User Interface Composition,“ in *Model-Driven Development of Advanced user Interfaces*, H. Hussmann, G. Meixner und D. Zuehlke, Hrsg., 2011, pp. pp. 43-65.

[67] „Mozilla Developer Network: Firefox,“ 2013. [Online]. Available: https://developer.mozilla.org/en-US/docs/Mozilla/Firefox/Releases. [Zugriff am 09 09 2013].

[68] „Mozilla Developer Network: XUL Tutorial,“ 2013. [Online]. Available: https://developer.mozilla.org/en-US/docs/XUL/Tutorial. [Zugriff am 09 09 2013].

[69] S. Crowle, „Personal Homepage,“ 25 05 2005. [Online]. Available: http://web.archive.org/web/20050119132150 http://dec.bournemouth.ac.uk/staff/scowle/ISML/Presentations/index.html. [Zugriff am 09 09 2013].

[70] F. Vanderdonckt, Q. Limbourg, B. Michotte, L. Bouillon, D. Trevisan und M. Florins, „USIXML: A User Interface Description Language for Specifying Multimodal User Interfaces,“ in *W3C Workshop on Multimodal Interaction*, Sophia Antipolis, 2004.

[71] Q. Limbourg, J. Vanderdonckt, B. Michotte, L. Bouillon und V. López-Jaquero, „USIXML: a Language Supporting Multi-Path Development of User Interfaces,“ in *Proceedings of Engineering Human Computer Interaction and Interactive Systems, Joint Working Conferences EHCI-DSVIS 2004*, Hamburg, Germany, July 11-13, 2004.

[72] H. Trætteberg, „Integrating Dialog Modelling and Application Development,“ in *Proceedings of the First International Workshop Conference on Making model-based UI design practical: usable and open methods and tools*, Funchal, Madeira, Portugal, 2004.

[73] F. Paternò, C. Santoro, J. Mäntyjärvi, G. Mori und S. Sansone, „Authoring Pervasive Multimodal user Interfaces,“ *International Jounal of Web Engineering and*

Technology, Volume 4 Issue 2, pp. pp. 235-261, May 2008.

[74] G. Mori, F. Paternò und C. Santoro, „Tool Support for Designing Nomadic Applications," in *IUI '03 Proceedings ofthe 8th international Conference on Intelligent User Interfaces*, New York, USA, 2003.

[75] „TERESA Tool," 2010. [Online]. Available: Downloadable from http://giove.isti.cnr.it/teresa.html. [Zugriff am 09 09 2013].

[76] „MARIAE Homepage," 2013. [Online]. Available: http://giove.isti.cnr.it/tools/MARIAE/home. [Zugriff am 08 12 2013].

[77] F. Paternò und C. Sisti, „Deriving vocal interfaces from logical descriptions in multi-device authoring environments," in *Proceeding of ICWE 2010 - 10th International Conference on Web Engineering*, 2010.

[78] J. Engel, C. Herdin und C. Märtin, „Evaluation of Model-based User Interface Development Approaches," in *Proceedings of HCII 2014, pp. 295-307*, Heraklion, Crete, 2014.

[79] J. Foley, „Transformations on a Formal Specification of User-computer Interfaces," *SIGGRAPH Comput. Graph., Vol. 21,* pp. 109-113, April 1987.

[80] J. D. Foley und P. Sukaviriya, „History, Results, and Bibliography of the User Interface Design Environment (UIDE), an Early Model-Based System for User Interface Design and Implementation," in *Fabio Paterno (Ed.), Interactive Systems: Design, Specification, and Verification, pp. 3-10*, Springer, Berlin, 1995.

[81] P. Sukaviriya und J. D. Foley, „Supporting Adaptive Interfaces in a Knowledge-based User Interface Environment," in *Proceedings of the 1st International Conference on Intelligent User Interfaces, IUI '93, pp. 107-113*, Orlando, Florida, USA, 1993.

[82] P. Sukaviriya, M. Frank, A. Spaans, T. Griffith, K. Bharat und J. Muthukumarasamy, „A Model-based User Interface Architecture: Enhancing a Runtime Environment with Declarative Knowledge," in *Fabio Paternò, ed., 'DSV-IS', pp. 181-197*, Springer, 1994.

[83] J. Foley, C. Gibbs, W. C. Kim und S. Kovacevic, „A Knowledge-based User Interface Management System," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 67-72*, Washington, D.C., USA, 1988.

[84] P. Sukaviriya, J. D. Foley und T. Griffith, „A Second Generation User Interface Design Environment: The Model and the Runtime Architecture," in *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, CHI*

'93, pp. 375-382, Amsterdam, The Netherlands, 1993.

[85] W. C. Kim und J. D. Foley, „DON: User Interface Presentation Design Assistant," in *Proceedings of the 3rd Annual ACM SIGGRAPH Symposium on User Interface Software and Technology, UIST '90, pp. 10-20*, Snowbird, Utah, USA, 1990.

[86] W. C. Kim und J. D. Foley, „Providing High-level Control and Expert Assistance in the User Interface Presentation Design," in *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, CHI '93, pp. 430-437*, Amsterdam, The Netherlands, 1993.

[87] P. Sukaviriya und J. D. Foley, „Coupling a UI Framework with Automatic Generation of Context-sensitive Animated Help," in *Proceedings of the 3rd Annual ACM SIGGRAPH Symposium on User Interface Software and Technology, UIST '90, pp. 152-166*, Snowbird, Utah, USA, 1990.

[88] W. E. Bennett, S. J. Boies, J. D. Gould, S. L. Greene und C. F. Wiecha, „Transformations on a Dialog Tree: Rule-Based Mapping of Content to Style," in *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software and Technology*, Williamsburg, Virginia, USA, 1989.

[89] C. Wiecha und S. Boies, „Generating User Interfaces: Principles and Use of ITS Style Rules," in *Proceedings of UIST'90*, 1990.

[90] C. Wiecha, W. Bennett, S. Boies, J. Gould und S. Greene, „ITS: A Tool for Rapidly Developing Interactive Applications," in *ACM Transactions on Information Systems*, 1990.

[91] P. Szekely, „Template-Based Mapping of Application Data to Interactive Displays," in *Proceedings of the 3rd Annual ACM Symposium on User Interface Software and Technology (UIST'90)*, Snowbird, Utah, USA, 1990.

[92] P. Luo, P. Szekely und R. Neches, „Management of Interface Design in HUMANOID," in *Proceedings of INTERCHI'93*, Amsterdam, Niederlande, 1993.

[93] P. Szekely, P. Luo und R. Neches, „Beyond Interface Builders: Model-Based Interface Tools," in *Proceedings of INTERCHI'93*, Amsterdam, Niederlande, 1993.

[94] P. Szekely, P. Luo und R. Neches, „Facilitating the Exploration of Interface Design Alternatives: The HUMANOID Model of Interface Design," in *Proceedings of SIGCHI'92*, 1992.

[95] P. Markopoulos, J. Pycock, S. Wilson und P. Johnson, „ADEPT - A Task Based Design Environment," in *Proceedings of the 25th Hawaii International Conference on System Sciences*, 1992.

[96] P. Johnson, S. Wilson, P. Markopoulos und J. Pycock, „ADEPT - Advanced Design Environment for Prototyping with Task Models," in *Proceedings of INTERCHI'93*, Amsterdam, Niederlande, 1993.

[97] S. Wilson und P. Johnson, „Bridging the Generation Gap: From Work Tasks to user Interface Designs," in *Computer-Aided Design of User Interfaces*, Namur, Belgien, 1996.

[98] C. Janssen, A. Weisbecker und J. Ziegler, „Generating User Interfaces from Data Models and Dialogue Net Specifications," in *Proceedings of InterCHI'93*, New York, USA, 1993.

[99] F. Bodart, A.-M. Hennebert, J.-M. Leheureux, I. Provor, B. Sacré und J. Vanderdonckt, „Towards a Systematic Building of Software Architecture: the TRIDENT Methodological Guide," in *Proceedings of Eurographics Workshop on Design, Specification, Verification of Interactive, pp. 237-253*, Toulouse, France, 1995.

[100] F. Bodart, A.-M. Hennebert, J.-M. Leheureux und J. Vanderdonckt, „A Model-Based Approach to Presentation: A Continuum from Task Analysis to Prototype," in *Proceedings of Eurographics Workshop on Design, Specification, Verification of Interactive Systems, pp. 25-39*, Bocca di Magra, Italy, 1994.

[101] F. Bodart und J. Vanderdonckt, „Widget standardisation through abstract interaction objects," in *Advances in Applied Ergonomics*, USA Publishing, 1996, pp. 300-305.

[102] C. Märtin und C. Winterhalder, „Integrating CASE and UIMS for Automatic Software Construction," in *Proceedings of the 5th Int. Conference on Human-Computer Interaction, HCI International '93, pp. 291-296*, Orlando, Florida, USA, 1993.

[103] C. Märtin, Modellierung, Entwurf und automatische Konstruktion interaktiver Softwaresysteme. Entwurf der modellbasierten Entwicklungsumgebung Application Modeling Environment (AME), Universität Rostock, Fakultät für Ingenieurwissenschaften, 1995.

[104] C. Märtin, „Software Life Cycle Automation for Interactive Applications: The AME Design Environment," in *Computer-Aided Design of User Interfaces, Proceedings of the 2nd International Workshop on Computer-Aided Design of User Interfaces, CADUI '96, pp. 57-74*, Namur, Belgium, 1996.

[105] C. Märtin, „Model-Based Software Engineering for Interactive Systems," in *Systems: Theory and Practice, Advances in Computing Science Series*, Springer, 1998, pp. 187-211.

[106] R. Neches, J. Foley, P. Szekely, P. Sukaviriya, P. Luo, S. Kovacevic und S. Hudson, „Knowledgeable Development Environments Using Shared Design Models," in *Proceedings of the 1st International Conference on Intelligent User Interfaces, IUI '93, pp. 63-70*, Orlando, Florida, USA, 1993.

[107] P. A. Szekely, P. N. Sukaviriya, P. Castells, J. Muthukumarasamy und E. Salcher, „Declarative Interface Models for User Interface Construction Tools: The MASTERMIND Approach," in *Proceedings of the IFIP TC2/WG2.7 Working Conference on Engineering for Human-Computer Interaction, pp. 120-150*, 1996.

[108] T. Browne, D. Dávila, S. Rugaber und K. Stirewalt, „The MASTERMIND User Interface Generation Project," in *GVU Technical Report, GIT-GVU-96-31*, 1996.

[109] T. Browne, D. Dávila, S. Rugaber und K. Stirewalt, „Using Declarative Descriptions to Model User Interfaces with MASTERMIND," in *Formal Methods in Human-Computer Interaction*, P. Palanque und F. Paternò, Hrsg., Springer London, 1998, pp. 93-120.

[110] R. E. K. Stirewalt, Automatic Generation of Interactive Systems from Declarative Models, Georgia Institute of Technology, Atlanta, GA, USA, 1997.

[111] H. Balzert, „Der JANUS-Dialogexperte: Vom Fachkonzept zur Dialogstruktur," *Softwaretechnik-Trends, herausgegeben von der Fachgruppe 'Software-Engineering' der GI,* August 1993.

[112] H. Balzert, „From OOA to GUI - the JANUS System," in *Proceedings of Human-Computer Interaction, INTERACT '95, IFIP TC13 Interantional Conference on Human-Computer Interaction, pp. 319-324*, Lillehammer, Norway, 1995.

[113] H. Balzert, F. Hofmann, V. Kruschinski und C. Niemann, „The JANUS Application Development Environment - Generating More than the User Interface," in *Computer-Aided Design of User Interfaces I, Proceedings of the Second International Workshop on Computer-Aided Design of User Interfaces*, Namur, Belgium, 1996.

[114] H. Balzert, „Das JANUS-System: Automatisierte Generierung von Benutzungsoberflächen," in *Lehrbuch Grundlagen der Informatik - Konzepte und Methoden in Java, C++ und UML, Algorithmik und Software-Technik*, Spektrum Akademischer Verlag Heidelberg Berlin, 1999, pp. 757-769.

[115] F. Lonczewski und S. Schreiber, „The FUSE-System: An Integrated User Interface Design Environment," in *Computer-Aided Design of User Interfaces*, Namur, Belgien, 1996.

[116] S. Schreiber, „Specification and Generation of User Interfaces with the BOSS-System," in *Proceedings of EWHCI'94, Volume 876 of Lecture Notes in Computer Sciences*, Berlin, 1994.

[117] S. Schreiber, „The BOSS System: Coupling Visual Programming with Model Based Interface Design," in *Interactive Systems: Design, Specification, and Verification - Focus on Computer Graphics*, Springer Berlin Heidelberg, 1995, pp. 161-179.

[118] A. R. Puerta , H. Eriksson, J. H. Gennari und M. A. Musen, „Beyond Data Models for Automated User Interface Generation," Proceedings of British HCI'94, pp. 353-366, 1994.

[119] A. R. Puerta, „The MECANO Project: Comprehensive and Integrated Support for Model-Based Interface Development," in *Computer-Aided Design of User Interfaces*, Namur University Press, 1996, pp. 19-36.

[120] T. Elwert und E. Schlungbaum, „Modelling and Generation of Graphical User Interfaces in the TADEUS Approach," in *Proc. of the 2nd Int. Eurographics Workshop on Design, Specification, and Verification of Interactive Systems DSV-IS'95 , pp. 193-208*, Toulouse, France, 1995.

[121] E. Schlungbaum und T. Elwert, „Dialogue Graphs: A Formal and Visual Specification Technique for Dialogue Modelling," in *Proceedings of the 1996 BCS-FACS Conference on Formal Aspects of the Human Computer Interface FAC-FA'96*, Sheffield, UK, 1996.

[122] E. Schlungbaum, „Individual User Interfaces and Model-based User Interface Software Tools," in *Proceedings of the 2nd International Conference on Intelligent User Interfaces IUI '97*, Orlando, Florida, USA, 1997.

[123] A. R. Puerta und D. Maulsby, „Management of Interface Design Knowledge with MODI-D," in *Proceedings of IUI'97, pp. 249-252*, Orlando, FL, USA, 1997.

[124] A. Puerta und J. Eisenstein, „Interactively Mapping Task Models to Interfaces in MOBI-D," in *Proceedings of Eurographics Workshop on Design, Specification and Validation of Interactive Systems (DSV-IS'98), pp.261-273*, 1998.

[125] A. Puerta und J. Eisenstein, „Towards a general computational framework for model-based interface development systems," in *Proceedings of the 4th International Conference on Intelligent User Interfaces (IUI '99), pp. 171-178*, Los Angeles, CA, USA,

1999.

[126] T. Griffiths, P. J. Barclay, J. McKirdy, N. W. Paton, P. D. Gray, J. Kennedy, R. Cooper, C. A. Goble, A. West und M. Smyth, „Teallach: A Model-Based User Interface Development Environment fo Object Databases," in *Proceedings of UIDIS'99*, Edinburgh, UK, 1999.

[127] „TADEUS Project Homepage," Johannes Kepler Universität Linz, Institut für Wirtschaftinformatik, Communications Engineering, [Online]. Available: http://www.ce.jku.at/de/projekte/[research_prototype]_TADEUS. [Zugriff am 08 08 2014].

[128] C. Stary, „TADEUS: Seamless Development of Task-based and User-oriented Interfaces," *Trans. Sys. Man Cyber. Part A,* pp. 509-525, September 2000.

[129] K. Gajos und D. S. Weld, „SUPPLE: Automatically Generating User Interfaces," in *IUI '04: Proceedings of the 9th International Conference on Intelligent User Interface*, New York, NY, USA, 2004.

[130] K. Z. Gajos, D. S. Weld und J. O. Wobbrock, „Automatically Generating Personalized User Interfaces with SUPPLE," *Artificial Intelligence Vol. 174,* pp. 910-950, 2010.

[131] K. Gajos und D. S. Weld, „Preference Elicitation for Interface Optimization," in *UIST '05: Proceedings of the 18th annual ACM Symposium on User Interface Software and Technology*, New York, USA, 2005.

[132] K. Z. Gajos, D. S. Weld und J. O. Wobbrock, „Decision-Theoretic User Interface Generation," in *AAAI'08*, AAAI Press, 2008, pp. 1532-1536.

[133] K. Z. Gajos, J. O. Wobbrock und D. S. Weld, „Automatically Generating User Interfaces Adapted to Users' Motor and Vision Capabilities," in *UIST '07: Proceedings of the 20th Annual ACM Symposium on User interface Software and Technology*, New Port, Rhode Island, USA, 2007.

[134] K. Gajos, D. Christianson, R. Hoffmann, T. Shaked, K. Henning, J. J. Long und D. S. Weld, „Fast and Robust Interface Generation for Ubiquitous Applications," *UbiComp 2005: Ubiquitous Computing, volume 3660 of Lecture Notes in Computer Science,* pp. 37-55, 2005.

[135] K. Z. Gajos, J. O. Wobbrock und D. S. Weld, „Improving the Performance of Motor-Impaired Users with Automatically-generated, Ability-Based Interfaces," in *CHI '08: Proceeding of the 26th Annual SIGCHI Conference on Human Factors in Computing*

Systems, New York, USA, 2008.

[136] F. Paternò, C. Santoro und L. D. Spano, „Engineering the Authoring of Usable Service Front Ends," in *The Journal of Systems and Software 84*, Elsevier Science Inc., New York, USA, 2011, pp. 1806-1822.

[137] F. Paternò, C. Santoro und L. D. Spano, „Support for Authoring Service Front-Ends," in *Proceedings of EICS'09 - 1st ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, Pittsburgh, PA, USA, 2009.

[138] F. Paternò, C. Santoro und L. D. Spano, „Exploiting Web Service Annotations in Model-based User Interface Development," in *Proceedings of EICS'10 - 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, Berlin, Deutschland, 2010.

[139] B. A. Myers, „State of the Art in User Interface Software Tools," in *Advances in Human-Computer Interaction, Volume 4*, R. H. Hartson und D. Hix, Hrsg., Norwood, NJ, USA, Ablex Publishing, 1992.

[140] P. Pinheiro da Silva, „User Interface Declarative Models and Development Environments: A Survey," in *DSV-IS'00 Proceedings of the 7th International Conference on Design, Specification, and Verification of Interactive Systems, pp. 207-226*, 2001.

[141] S. Berti, F. Correani, G. Mori, F. Paternò und C. Santoro, „TERESA: A Transformation-based Environment for Designing and Developing Multi-device Interfaces," in *CHI '04 Extended Abstracts on Human Factors in Computing Systems, pp. 793-794*, Vienna, Austria, 2004.

[142] F. Paternò, C. Santoro und L. D. Spano, „User Task-based Development of Multi-device Service-oriented Applications," in *Proceedings of the International Conference on Advanced Visual Interfaces, LNCS, Vol. 5726*, Roma, Italy, 2010.

[143] A. Seffah, „The Evolution of Design Patterns in HCI: From Pattern Languages to Pattern-oriented Design," in *Proceedings of the1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems (PEICS '10), pp. 4-9*, Berlin, Germany, 2010.

[144] C. Alexander, The Timeless Way of Building, New York: Oxford University Press, 1979.

[145] C. Kruschitz und M. Hitz, „The Anatomy of HCI Design Patterns," in *Proceedings of the 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, pp. 202-207*, Athens, Greece, 2009.

[146] C. Kruschitz und M. Hitz, „Human-Computer Interaction Design Patterns: Structure, Methods, and Tools," *International Journal on Advances in Software, Vol. 2, no. 1&2,* pp. 225-237, 2010.

[147] D. van Duyne, J. Landay und J. Hong, „The Design of Sites, Patterns for Creating Winning Websites," 2006.

[148] A. Cockburn, A. Baruz, A. Engelund, B. Hanes Perry, C. Brown, C. Siska, D. Olson, E. Poor, G. Xexeo, I. Lowe, J. Chapman, J. Coplien, J. Holloway, K. Brown, M. Eichin, R. Phillips, R. Jeffries, S. Gordon und S. McCormick, „Anti Pattern," 21 11 2012. [Online]. Available: http://c2.com/cgi/wiki?AntiPattern. [Zugriff am 25 02 2015].

[149] W. J. Brown, R. C. Malveau, H. W. McCormick,III und T. J. Mowbray, AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, New York, NY, USA: John Wiley & Sons, Inc., 1998.

[150] P. Kotzé, K. Renaud, K. Koukouletsos, B. Khazaei und A. Dearden, „Patterns, Antipatterns and Guidelines - Effective Aids to Teaching Principles?," in *Inventivity: Teaching Thoery, Design, and Innovation in HCI - Proceedings of HCIEd2006-1 (First Joint BCS / IFIP WG 13.1 / ICS/EU CONVIVIO HCI Educators Workshop), pp. 115-120*, University of Limerick, Ireland, 2006.

[151] J. O. Coplien, „Software Design Patterns: Common Questions and Answers," in *The Patterns Handbook: Techniques, Strategies, and Applications*, Cambridge University Press, New York, NY, USA, 1998, pp. 311-320.

[152] A. Tešanović, „What is a Pattern?," Department of Computer and Information Science, Linköping University, Linköping, Sweden, 2001.

[153] B. Appleton, „Patterns and Software: Essential Concepts and Terminology," 14 02 2000. [Online]. Available: http://csis.pace.edu/~grossman/dcs/Patterns%20and%20Software-%20Essential%20Concepts%20and%20Terminology.pdf. [Zugriff am 25 02 2015].

[154] C. Alexander, S. Ishikawa und M. Silverstein, A Pattern Language, New York: Oxford University Press, 1977.

[155] C. Alexander, The Oregon Experiment, New York: Oxford University Press, 1975.

[156] E. L. Hutchins, J. D. Hollan und D. A. Norman, „Direct Manipulation Interfaces," in *User Centered System Design - New Perspectives on Human-Computer Interaction*, D. A. Norman und S. W. Draper, Hrsg., Hillsdale, New Jersey, USA, Lawrence Erlbaum

Associates Inc., Publishers, 1986, pp. 87-124.

[157] K. Beck und W. Cunningham, „Using Pattern Languages for Object Oriented Programs," in *Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, 1987.

[158] T. Coram und J. Lee, „Experiences - A Pattern Language for User Interface Design," 1996. [Online]. Available: http://www.maplefish.com/todd/papers/Experiences.html. [Zugriff am 24 11 2013].

[159] A. Marcus, „Patterns within Patterns," in *Interactions, Vol. 11, Issue 2, pp. 28-34*, 2004.

[160] T. Tiedke, T. Krach und C. Märtin, „Multi-Level Patterns for the Planes of User Experience," in *Proceedings of HCI International 2005, pp. 1159-1167*, Las Vegas, Nevada, USA, 2005.

[161] J. Guerrero, J. Vanderdonckt, J. M. González Calleros und M. Winckler, „Towards a Library of Workflow User Interface Patterns," in *Proc. of 15th Int. Workshop on Design, Specification, and Verification of Interactive Systems (DSV-IS '08), pp. 96-101*, Kingston, Canada, 2008.

[162] A. Gaffar, D. Sinnig, A. Seffah und P. Forbrig, „Modeling Patterns for Task Models," in *Proceedings of the 3rd Annual Conference on Task Models and Diagrams (TAMODIA '04), pp. 99-104*, Prague, Czech Republic, 2004.

[163] J. Yoder und J. Barcalow, „Architectural Patterns for Enabling Application Security," in *Fourth Conference on Patterns Languages of Programs (PLoP '97)*, Monticello, Illinois, USA, 1997.

[164] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad und M. Stal, Pattern-Oriented Software Architecture - A System of Patterns, John Wiley & Sons, New York, USA, 1996.

[165] D. Riehle und H. Züllighoven, „Understanding and Using Patterns in Software Development," *Theory and Practice of Object Systems - Special issue on patterns, Vol. 2, Issue 1,* p. 3.13, 1996.

[166] M. van Welie und G. C. van der Veer, „Pattern Languages in Interaction Design: Structure and Organization," in *Proceedings of Human-Computer Interaction - INTERACT'03, pp.527-534*, 2003.

[167] A. Kubo, H. Washizaki und Y. Fukazawa, „A Metric for Measuring the Abstraction Level of Design Patterns," in *Proceedings of the 14th Conference on Pattern Languages of Programs (PLOP '07)*, Monticello, Illinois, USA, 2007.

[168] J. Noble, „Classifying Relationships Between Object-Oriented Design Patterns," in *Proceedings of the Australian Software Engineering Conference (ASWEC '98), pp. 98-107*, Adelaide, South Australia, 1998.

[169] M. Völter, „Server-Side Components - A Pattern Language," in *Proceedings of the 6th European Conference on Pattern Languages of Programms (EuroPLoP '01)*, Irsee, Germany, 2001.

[170] W. Zimmer, „Relationships Between Design Patterns," in *Pattern Languages of Program Design*, J. O. Coplien und D. C. Schmidt, Hrsg., New York, NY, USA, ACM Press/Addison-Wesley Publishing Co., 1995, pp. 345-364.

[171] M. Taleb, H. Javahery und A. Seffah, „Pattern-Oriented Design Composition and Mapping for Cross-Platform Web Applications," in *Proceedings of DSV-IS '06*, Dublin Ireland, 2006.

[172] J. Engel, C. Herdin und C. Märtin, „Exploiting HCI Pattern Collections for User Interface Generation, pp. 36-44," in *Proceedings of PATTERNS 2012, IARIA*, Nice, France, 2012.

[173] S. Fincher und J. Finlay, „Perspectives on HCI Patterns: Concepts and Tools (Introducing PLML)," *Interfaces, Vol. 56,* pp. 26-28, 2003.

[174] J. Deng, E. Kemp und E. G. Todd, „Focusing on a standard pattern form: the development and evaluation of MUIP," in *CHINZ '06: Proceedings of the 7th ACM SIGCHI New Zealand Chapter's International Conference on Compter-Human Interaction, pp.83-90*, Christchurch, New Zealand, 2006.

[175] D. Bienhaus, „PLMLx Doc," 2004. [Online]. Available: http://www.cs.kent.ac.uk/people/staff/saf/patterns/diethelm/plmlx_doc/index.html. [Zugriff am 24 11 2013].

[176] C. Kruschitz, „XPLML - A HCI Pattern Formalizing and Ubifying Approach," in *Proceedings of CHI EA '09*, 2009.

[177] J. Dietrich und C. Elgar, „A Formal Description of Design Patterns Using OWL," in *Proceedings of Software Engineering Conference*, 2005.

[178] C. Alexander, „The Origins of Pattern Theory: The Future of the Theory, and the Generation of a Living World," in *IEEE Software (Volume 16, Issue 5)*, 1999.

[179] ObjectVenture Inc., „Pattern and Component Markup Language (PCML) Draft 3," 2002. [Online]. Available: http://www.cryer.co.uk/glossary/p/pcml/PCMLSpecification.pdf. [Zugriff am 26 12 2013].

[180] D. Sinnig, The Complicity of Patterns and Model-based UI Development, Master Thesis, Concordia University, Montreal, Québec, Canada, 2004.

[181] J. Tidwell, Designing Interfaces, Patterns for Effective Interaction Design, Bd. 2nd Edition, O'Reilly Media Inc., 2011.

[182] J. Tidwell, „Designing Interfaces, Patterns for Effective Interaction Design (Second Edition)," [Online]. Available: http://www.designinginterfaces.com/patterns/. [Zugriff am 27 12 2014].

[183] M. van Welie, „Patterns in Interaction Design," 2008. [Online]. Available: http://www.welie.com. [Zugriff am 24 11 2013].

[184] D. van Duyne, J. Landay und J. Hong, „The Design of Sites, Patterns for Creating Winning Websites," 2009. [Online]. Available: http://www.designofsites.com/design-patterns/. [Zugriff am 24 11 2013].

[185] „Yahoo Design Pattern Library," Yahoo! Inc., [Online]. Available: https://developer.yahoo.com/ypatterns/. [Zugriff am 24 11 2014].

[186] „Quince UX Patterns Explorer," Infragistics Inc., [Online]. Available: http://quince.infragistics.com/#/Main. [Zugriff am 07 12 2014].

[187] J. Deng, E. Kemp und E. G. Todd, „Managing UI Pattern Collections," in *Proceedings of CHINZ '05, pp. 31-38*, Auckland, New Zealand, 2005.

[188] N. Li, Usability patterns-assisted design for Web user interfaces, Master Thesis, Concordia University, Montreal, Québec, Canada, 2001.

[189] A. Ansari, UPADE : A Tool for Automating HCI Pattern-oriented Designs, Master Thesis, Concordia University, Montreal, Québec, Canada, 2003.

[190] J. Lin und J. A. Landay, „Damask: A Tool for Early-Stage Design and Prototyping of Multi-Device User Interfaces," in *Int. Conf. Distributed Multimedia Systems Workshop on Visual Computing*, San Francisco, CA, USA, 2002.

[191] J. Lin und J. A. Landay, „Employing Patterns and Layers for Early-stage Design and Prototyping of Cross-Device User Interfaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'08), pp. 1313-1322*, Florence, Italy, 2008.

[192] A. Gaffar, D. Sinnig, H. Javahery und A. Seffah, „MOUDIL: A Comprehensive Framework for Disseminating and Sharing HCI Patterns," in *CHI 2003 workshop on HCI Patterns: Concepts and Tools*, Fort Lauderdale, Florida, USA, 2003.

[193] A. Gaffar, A. Seffah und J. A. Van der Poll, „HCI Pattern Semantics in XML: A Pragmatic Approach," *SIGSOFT Softw. Eng. Notes, Vol. 30,* pp. 1-7, May 2005.

[194] A. Gaffar und A. Seffah, „An XML Multi-Tier Pattern Dissemination System," in *Encyclopedia of Database Technologies and Applications*, L. C. Rivero, J. H. Doorn und V. E. Ferraggine, Hrsg., Idea Group, 2005.

[195] A. Roski, Entwicklung einer HCI-Pattern-Language für Websites unter Nutzung von Ergebnissen der UX-Forschung, Fachhochschule Augsburg, Deutschland, 2006.

[196] R. O. Badr und H. M. Hosny, „An HCI Pattern Language Management Tool," in *Proceedings of the 2006 International Conference on Computer Engineering and Systems, pp. 337-342*, Cairo, Egypt, 2006.

[197] A. Seffah und A. Gaffar, „Model-based User Interface Engineering with Design Patterns," *Journal of Systems and Software, Vol. 80, Issue 8,* pp. 1408-1422, August 2007.

[198] B. Weyers, J. Nanninga und W. Luther, „Collaborative Creation of Interaction Patterns for the Use in User Interface Generation," in *7th International Conference on Collaboration Technologies (CollabTech 2014)*, Santiago, Chile, 2014.

[199] J. Tidwell, „Designing Interfaces, Patterns for Effective Interaction Design (First Edition)," [Online]. Available: http://designinginterfaces.com/firstedition/. [Zugriff am 27 12 2014].

[200] J. Engel, C. Märtin und P. Forbrig, „Practical Aspects of Pattern-supported Model-driven User Interface Generation," in *Proceedings of HCI International 2017, Vol. I, Springer LNCS, pp. 397-414*, Vancouver, Canada, 2017.

[201] A. Fesenmeier, Entwicklung eines Pattern-Repository zur Verwaltung von Entwurfsmustern für die automatische Generierung von Benutzeroberflächen, Masterarbeit an der Fakultät für Informatik, Hochschule Augsburg, 2015.

[202] R. Reitböck, Modell- und musterbasierte Entwicklung interaktiver Benutzeroberflächen für Android Applikationen, Masterarbeit an der Fakultät für Informatik, Hochschule Augsburg, 2015.

[203] C. Märtin, J. Engel, A. Fesenmeier, R. Reitböck und C. Herdin, „Werkzeugunterstützung für die Pattern- und modellgetriebene Entwicklung interaktiver Systeme in PaMGIS 2.0," in *Forum Medientechnik - Next Generation, New Ideas, 26.-27. November 2015, Seidl, M. und Schmiedl, G. (eds.), vwh, pp. 111-124*, St. Pölten, Österreich, 2015.

[204] C. Märtin, C. Herdin und J. Engel, „Patterns and Models for Automated User Interface Construction - In Search of the Missing Links," in *Human Computer Interaction, Part I, LNCS 8004, pp. 401-410*, Las Vegas, Nevada, USA, 2013.

[205] SELFHTML e.V., „SELFHTML," 2007. [Online]. Available: http://de.selfhtml.org/css/. [Zugriff am 01 12 2013].

[206] W3C, „W3C Working Group Note: XBL 2.0," 24 05 2012. [Online]. Available: http://www.w3.org/TR/xbl/. [Zugriff am 01 12 2013].

[207] Apple, „OS X," 2013. [Online]. Available: http://www.apple.com/de/osx/. [Zugriff am 01 12 2013].

[208] IBM Corp., „OS/2 Warp," 2013. [Online]. Available: http://www-01.ibm.com/software/os/warp-withdrawal/. [Zugriff am 01 12 2013].

[209] Oracle Corp., „Solaris," 2013. [Online]. Available: http://www.oracle.com/us/products/servers-storage/solaris/overview/index.html. [Zugriff am 01 12 2013].

[210] IBM Corp., „AIX," 2013. [Online]. Available: http://www-03.ibm.com/systems/power/software/aix/about.html. [Zugriff am 01 12 2013].

[211] Heise Online, „HP deutet Ende von OpenVMS an," 2013. [Online]. Available: http://www.heise.de/ix/meldung/HP-deutet-Ende-von-OpenVMS-an-1886348.html. [Zugriff am 01 12 2013].

[212] Microsoft Corp., „Windows-Produkte," 2013. [Online]. Available: http://windows.microsoft.com/de-de/windows/products. [Zugriff am 01 12 2013].

[213] P. Markopoulos, A Compositional Model for the Formal Specification of User Interface Software, PhD Thesis, Queen Mary and Westfield College University of

London, 1997.

[214] P. Forbrig, Objektorientierte Softwareentwicklung mit UML, M. Lutz und C. Märtin, Hrsg., Hanser-Verlag München, 2007, pp. 124-145.

[215] M. E. Modell, „Entity-Relationship-Attribute Modeling," 2007. [Online]. Available: http://www.martymodell.com/ddsd/ddsd20.html. [Zugriff am 03 07 2014].

[216] „UsiXML Homepage," [Online]. Available: http://usixml.org. [Zugriff am 07 12 2014].

[217] „DiaMODL Project," [Online]. Available: http://sourceforge.net/projects/diamodl/. [Zugriff am 07 12 2014].

[218] R. Pausch, M. Conway und R. Deline, „Lessons Learned from SUIT, the Simple User Interface Toolkit," *ACM Trans. Inf. Syst., vol.4,* pp. 320-344, 1992.

[219] P. Johnson, H. Johnson, R. Waddington und A. Shouls, „Task-Related Knowledge Structures: Analysis, Modelling and Application," in *People and Computers IV*, University Press, Cambridge, UK, 1988, pp. 35-62.

[220] C. A. R. Hoare, „Communicating Sequential Processes," *Communication of the ACM, Vol. 21, Issue 8,* pp. 666-677, August 1978.

Anhang

A. Vergleich der untersuchten UIDL im Detail

Tabelle 40 ERGEBNISSE UIDL-VERGLEICH – TEIL 1

UIDL	Originator	Start of Developm. / First Publication	Current Version / Latest Publication	Target
UIML	Virginia Polytechnic Institute and State University, Harmonia Inc. [40]	Start of development 1997 [40] 1998 (UIML 1.0) [41]	4.0 Committee Draft (OASIS) (23.02.2008) [41]	Multi-platform [40] [41]
XUL	Open-Source-Projekt Mozilla [42]	1998 [42]	In context of Firefox 17 (2012) [67]	Multi-platform [68]
XIML	RedWhale Software [46] [48]	1999 [47] XIML 1.0 [46]	Latest publication 2002 [46]	Multi-platform [46] [48] Multi-user [46]
ISML	Bournemouth University [49]	Proposal 2000 [69]	Latest publications 2003 [49] [50]	Multi-platform [52]
UsiXML	CAMELEON FP5 Project [53]	2003 [52]	UsiXML V1.8 (2007) [54]	Multi-platform [52]
DiaMODL	Norwegian Univer-sity of Science and Technology [72]	First publication 2003 [57]	Lastest publication 2008 [56]	Multi-platform [57]
TERESA XML	CNR-ISTI, HIIS Laboratory, Pisa [59]	2003 [60]	Version 3.4 (2009) [59] Predecessor version of MARIA [60]	Multi-platform [61] [62]
MARIA XML	CNR-ISTI, HIIS Laboratory, Pisa [76]	Successor of TERESA XML [60] 2009 [64]	MARIA V1.4.12 (2011) [60] MARIAE 1.5.4 (2013) [76]	Multi-platform [77]

Tabelle 41 ERGEBNISSE UIDL-VERGLEICH – TEIL 2

UIDL	Models	Methodology	Major Concepts
UIML	Presentation, Dialog, Domain [41]	XML-based specification of multiple UI presentations [41], one UIML document is needed for each UI toolkit [41]	4 components: head, interface, peers, template interface: Structure, style, content, behavior [40]
XUL	Presentation, Dialog [38]	Specification of a generic UI description [52]	Window, Box Model [44]
XIML	Task, Domain, User, Presentation, Dialog [46] [48]	Specification of a generic UI description [52]	Strict separation of UI definition and its actual display on a target device [48]; Components, Relations, Attributes [46] [48]; Elements, Statements, Definitions [46]
ISML	Task, Metaphor [49], Presentation, Domain, Dialog [52], Device [50]	Specification of a generic UI description [52]	Make metaphors part of the model-based design. Decoupling the metaphor model from specific implementation details [49]
UsiXML	Task, Domain, Presentation, Dialog, Context of Use (User, Platform, Environment) [51] [71], Mapping, Translation [55], UI [70] [71]	Specification of generic UI description [52] according to CAMELEON Reference Framework [53]	4 levels of abstraction: Task&Concepts, AUI, CUI, FUI [70]
DiaMODL	Domain [57] Dialog [57] [72]	UML-based, combines a dataflow-oriented language with UML Statecharts [57]	Extending UML Meta-model: Interactors receive and send information through gates (Input/Send, Output/Receive, Input/Receive, Output/Send) [57]
TERESA XML	Task [61] [62], Domain, Dialog, Presentation, Device [61]	Incremental model-based design: high-level task and domain model (CTT), platform-dependent system task model, abstract user interface (AUI), UI generation [61], Models stored in XML format [58]	Presentation task sets (PTS), Interactor-based description (AUI): Presentation (static structure), Dialog (dynamic behavior) Concrete UI, Final UI for target platform [62]
MARIA XML	Task, Domain [60], Presentation, Event [64], Dialog [66]	XML-based [60] specification of generic UI description Task modeling by means of ConcurTaskTrees [66] UML 2.0 class diagrams [60]	Task Level (CTT), Abstract UI Level, Concrete UI Level [66]

Tabelle 42 ERGEBNISSE UIDL-VERGLEICH – TEIL 3

UIDL	Supported Languages	Supported Platforms	Tool Support
UIML	Java, HTML, WML, VoiceXML [40] C++, CORBA, QT [52]	Handheld, Desktop, Mobile phone, TV [52]	4 tools[198]
XUL	XUL [38]	BSD, Linux, OS X, Solaris, OS/2, AIX, OpenVMS, Windows [45]	6 tools[199]
XIML	C++, HTML, WML [46] Java Swing [52]	Desktop PC, PDA, Cell Phone [48]	2 tools[200]
ISML	Java [52]	Desktop PC, 3D Screen [52]	ISML Framework [49]
UsiXML	HTML, XHTML, VoiceXML, Java3D, VRML, X3D, XAML, Java, Flash, QTK, WML, X+V, C++ [52]	Mobile, PocketPC, Interactive Kiosk, Wall screen, PDA [52]	13 tools[201]
DiaMODL	Java Swing [72] Java, CORBA [57]	Not specified	Eclipse[202]
TERESA XML	XHTML [61] C# [73]	Desktop PC, Mobile phone, VoiceXML [61] [62], Digital TV, X+V [73]	TERESA, CTTE [61] [74]
MARIA XML	HTML 4/5, JSP, VoiceXML, X+V, SMIL [60]	Destop PC, Mobile [60], Vocal [77]	CTTE [66] MARIAE 1.5.6 [76]

[198] Transformation-based Integrated Development Environment (TIDE), LiquidUI product suite (Harmonia) [40], VoiceXML Renderer, WML Renderer [52]
[199] Gecko, XPCom, XPConnect, XPInstall, XULRunner, XUL Explorer [43]
[200] MOBI-D Model-to-XIML Converter, HTML-to-XIML Converter [46]
[201] UsiGesture V1.0 (Editor), FlashiXML [216], SketchiXML, IdealXML, VisiXML, KnowiXML, TransformiXML [55], FlowiXML, QTKiXML, InterpiXML, Attributed Graph Grammars (AGG) tool [71], GrafiXML, ReversiXML [55] [71]
[202] Eclipse-based editors, views and runtime [217]

Tabelle 43 ERGEBNISSE UIDL-VERGLEICH – TEIL 4

UIDL	Availability	Number of Tags	Abstraction Level
UIML	Open specification without license agreement [41]	44 [41]	Model [41]
XUL	Requires open source rendering engine Gecko [45]	125 [44]	CUI [44]
XIML	Free XIML Research License Agreement [47]	32 [52]	Model [46] AUI, CUI [48]
ISML	Not specified	Not specified	Model
UsiXML	Free to use [53]	118 Tags [52] vs. 345 Tags [54]	Model, CUI, AUI, FUI [55]
DiaMODL	Uses open source tools, UML, MOF/MXI [57]	Not specified	Model
TERESA XML	Free download of TERESA (incl. TERESA XML) [59]	AUI (38) CUI platform-dependent, e.g. Desktop (86), DTV (67), Voice (66), Mobile (80) [75]	Model, AUI, CUI [61]
MARIA XML	Free download of MARIAE (incl. MARIA XML) [76]	Not specified	Model, CUI [60] [66] AUI [66]

B. Vergleich der untersuchten MB-UIDE im Detail

Tabelle 44 ERGEBNISSE MB-UIDE-VERGLEICH – TEIL 1

MB-UIDE	Originator	Start of Dvlpm. / First Publication	Current Version / Latest Public.	Functionality
UIDE	The George Washington University [79]	1987 [79] [80]	1995 [80]	Modeling, Generation, Runtime [80] [81] [82] [84]
ITS	IBM T.J. Wastson Research Center, Yorktown Heights [89] [90]	1989 [88]	1990 [89] [90]	Modeling, Runtime [89]
HUMANOID	University of Southern California (USC), Los Angeles [91]	1990 [91]	1993 [93]	Modeling, Runtime [92] [93]
ADEPT	Queen Mary and Westfield College, University of London [96] [97] [95]	1992 [95]	1996 [97]	Modeling, Generation, Runtime [95]
GENIUS	Fraunhofer Institut für Arbeitswirtschaft und Organisation, Stuttgart [98]	1993 [98]	1993 [98]	Modeling, Generation [98]
TRIDENT	Facultés Universitaires Notre-Dame de la Paix, Namur [24]	1993 [24]	1996 [101]	Modeling, Generation [100]
AME	Fachhochschule Augsburg [102] [104]	1993 [102]	1998 [105]	Modeling, Generation [102] [103], Runtime [102]
MASTERMIND	Georgia Institute of Technology, Atlanta and University of Southern California, Los Angeles [106] [107] [108] [109]	1993 [106]	1998 [109]	Modeling, Runtime [106] [107], Generation [107]
JANUS	Ruhr-Universität Bochum [112] [113]	1993 [111]	1999 [114]	Modelling [112] [113] [114]
FUSE	Technische Universität München [115]	FUSE 1996 [115] BOSS 1994 [117]	1996 [115]	Modeling, Generation, Runtime [115]
MECANO	Stanford University [118] [119]	1994 [118]	1996 [119]	Modeling, Generation, Runtime [118]
TADEUS (Rostock)	Universität Rostock [120] [121] [22]	1995 [120]	1997 [122]	Modeling, Generation [120] [21]
MOBI-D	Stanford University [119] [123] [124] [125]	1997 [123]	1999 [125]	Modeling, Generation, Runtime [124]
TEALLACH	University of Manchester, Napier University and University of Glasgow [126]	1999 [126]	1999 [126]	Modeling, Generation [126]
TADEUS (LINZ)	Universität Linz [128]	2000 [128]	2000 [128]	Modeling, Generation,Runtime [128]

MB-UIDE	Originator	Start of Dvlpm. / First Publication	Current Version / Latest Public.	Functionality
TERESA	Istituto di Scienza e Tecnologie dell‘ Informazione (ISTI) – Consiglio Nazionale delle Ricerche (CNR) [58] [73] [74] [141]	2003 [74]	2008 [73]	Modeling, Generation [62]
SUPPLE (++)	University of Washington, Seattle [129]	2004 [129]	2010 [130]	Modeling, Generation [129] [134], Runtime [129] [134] [131]
MARIAE	Istituto di Scienza e Tecnologie dell‘ Informazione (ISTI) – Consiglio Nazionale delle Ricerche (CNR) [64] [65] [137] [138] [142] [136]	2009 [64] [65] [137]	Current version: MARIAE 1.5.6 [76] Latest publication 2011 [136]	Modeling, Generation [64] [65]

Tabelle 45 ERGEBNISSE MB-UIDE-VERGLEICH – TEIL 2

MB-UIDE	Models	Model Notations	Abstraction Levels	Target
UIDE	User Model [81] [84] [80], Application Model, Interface Model [82] [80], Data Model [84], Operational Model [80], Knowledge Model [79] [81] [82]	Pascal-like syntax [79], C++ code [84], Interface Description Language (IDL) [80]	Model, AUI, CUI [81] [84]	Multi-platform [82] [84] [86], Multi-user [81]
ITS	Data pool (domain model) [89], Dialog Model [88]	Proprietary notations [90] [88]	Model, AUI, CUI [88]	Multi-platform, Multi-user [89]
HUMANOID	Application Model (enriched) [93]	Not specified	Model, AUI, CUI [93]	Multi-platform [91]
ADEPT	User Model, Task Model [96] [95] (existing and envisioned Task Model [97]), Dialog Model, Interface Model [95]	Task Model: Task Knowledge Structures (TKS) [95] [96], Dialog Model: Communication Sequential Processes (CSP) [95]	Model, AUI, CUI [95]	Multi-platform (additional generators required) [96]
GENIUS	Data Model, Views (Dialog Model) [98]	Data Model: Entity Relationship Model (ERM), Views (dynamic aspects): Dialog Nets [98]	Model, AUI [98]	Multi-platform [98]

MB-UIDE	Models	Model Notations	Abstraction Levels	Target
TRIDENT	Task Model (enriched) [99] [100], Data Model [24] [99] [100], Dialog Model [23] [24], User Model [23] [24] [100]	Enriched Task Model: Activity Chaining Graph (ACG) [100] [99], Data Model: Entity Relationship Attribute (ERA) Model [23], Dialog Model: State-Transition Diagram [100]	Model, AUI, CUI, FUI [100] [24] [23]	Multi-platform [23] [24], Multi-user, Multi-environment [23]
AME	Domain Model, Application Model, User Interface Model [102] [103] [104]	Domain Model: OOA-Model [102] [104], Application Model, User Interface Model: OOA-/OOD-Model [102] [103] [104]	Model, AUI, CUI, FUI [102] [103] [104]	None [102]
MASTERMIND	Application Model, Task Model, Presentation Model [107] [108] [109] [110], Dialogue Model, Interaction Model, Context Model [108] [109]	Common Object Request Broker Architecture (CORBA) [107], Task Model: MASTERMIND Dialogue Language (MDL) [110], Presentation, Dialog, Interaction Models: MASTERMIND Textual Format (MTF) [108] [109]	Model, CUI [108] [109] [110], FUI [108] [110]	Multi-platform [107] [108] [109]
JANUS	Problem Domain Model [112] [114]	OOA Model specified in JANUS Definition Language (JDL) [112] [114]	Model [112] [113] [114], AUI [113], CUI [114], FUI [113] [114]	Multi-platform [113] [114]
FUSE	Task Model, Problem Domain Model, User Model, Formal logical user interface specification [115]	Problem Domain Model: algebraic specifications of functions and data structures, Logical UI: Hierarchic Interaction Graph Templates (HIT) [115]	Model, AUI, CUI [115]	Multi-layout (switch at runtime) [115]
MECANO	Domain Model [118] [119], Interface Model [118]	Frame-based representation language defining class hierarchies [118]	Model, AUI, CUI [118]	Multi-platform [118]
TADEUS (Rostock)	Problem Domain Model, Task Model, Dialog Model, User Model [120] [22] [122] [21]	Domain Model: OOA Object Model [120], Dialog Model: Dialogue Graph [120] [121] [22]	Model, AUI, CUI [22] [120], FUI [120]	Multi-platform [21], Multi-user [122]

MB-UIDE	Models	Model Notations	Abstraction Levels	Target
MOBI-D	Domain Model, User Task Model, Dialog Model Design Model, Presentation Model, User Model [123] [124] [125], Generic interface models called Mecano Interface Models (MIM) [123]	MIMIC [123] [124]	Model, AUI, CUI Model [123] [124] [125]	Multi-platform, Multi-user [123]
TEALLACH	Domain Model, Task Model, Presentation Model [126]	Domain Model: concepts specified in ODMG object database standard, Task Model: goal-oriented task hierarchy [126]	Model, AUI, CUI [126]	Multi-platform (platform independent) [126]
TADEUS (LINZ)	Business Intelligence Model [128], Problem Domain Model, Task Model, Interaction Model, User Model [127] [128]	OOA diagrams: Object Relationship Diagrams (ORD), Object Behavior Diagrams (OBD), Object Interaction Diagrams (OID)	Model, AUI, CUI [128]	Multi-platform, Multi-user [128]
TERESA	Domain Model, Task Model, System Task Model [74], Interaction Model [73]	Task Model: CTT [74], Abstract User Interface (AUI) specification: TERESA XML [73]	Model, AUI, CUI, FUI [73] [74] [62]	Multi-platform [74] [73], Multi-modal [73]
SUPPLE (++)	Interface Model, Data Model (included in Interface Model), Device Model, User Model [129] [134], Cost Model (included in Device Model) [134], Preference Model (SUPPLE), Ability Model (SUPPLE++) [135]	Interface Model: set of interface elements and set of interface constraints [129] Device Model: set of UI widgets, device-specific constraints, cost functions [129] User Model: user traces [129]	Model, AUI, CUI [129]	Multi-device [129], Multi-user [129] [133]
MARIAE	Task Model [64] [65] [136] [142], Data Model [64] [65] [137] [142], Event Model [64] [65], Dialog Model [64] [65] [142]	Task Model: CTT [64], Data Model: XSD type definition [64] [65]; AUI/CUI: MARIA XML [138], Transformations: XSLT [65]	Model, AUI, CUI, FUI [65]	Multi-platform [64] [65] [142]

Tabelle 46 ERGEBNISSE MB-UIDE-VERGLEICH – TEIL 3

MB-UIDE	Target Languages	Target Platforms	Major Concepts
UIDE	GWUIMS [79], OLIT, Motif[203] [81] [82], C++ [82]	Frame-based systems [79]	Mapping between Application Model and Interface Model [80] [82], Objects, actions on objects, pre- and post-conditions on actions [80] [83], automatic layout of dialog boxes [80], automatic generation of context-sensitive help [80], automatic dialog sequencing [80] [84], automatic evaluation of UI [80]
ITS	ITS Runtime Environment [88]	Not specified	Separation of application style from content [88], 4-laver architecture (action, dialog, style rule, style program) [90], Application design and development process with 4 separate roles (application expert, application programmer, style expert, style programmer), Style Implementation Routine (SIR) [88]
HUMANOID	HUMANOID Runtime Environment [93]	Not specified	5 design dimensions (application design, presentation, manipulation, sequencing, action side-effects) [94], Incremental modeling, Part replication [93], Exploration of design alternatives, Usage of defaults [92], Templates [91]
ADEPT	OpenLook [96], support of further target languages [96]	Not explicitly specified, but OpenLook is a GUI specification for Unix systems[204]	Existing task model, Envisioned task model, Design guidelines [97], Abstract Interface Model [95] [96], Concrete Interface Model (CIM) [96]
GENIUS	Existing UIMS (name not specified) [98]	Database-oriented applications [98]	Views defined for a data model, Explicit design rules for creating static UI layout, Dialog nets for specification of dynamic behavior [98]

[203] For details see e.g. http://www.opengroup.org/motif/
[204] See e.g. http://www.oreilly.com/openbook/openlook/

MB-UIDE	Target Languages	Target Platforms	Major Concepts
TRIDENT	Simple User Interface Toolkit[205] (SUIT) [100], OSF/Motif[203] [23]	Not explicitly specified, but SUIT is available on Macintosh, DOS, and Unix platforms[205] and OSF/Motif on Unix[203], Microsoft Windows [99]	Separation of conversation and presentation [100], Abstract Interaction Objects (AIO), Concrete Interaction Objects (CIO) [23] [24] [99] [100], Rule-based automatic generation [23] Decision Trees [24]
AME	C++, KAPPA-PC Application Language (KAL), OSF/Motif[203] [102]	Windows [102] [103] [104], UNIX [102]	UI Generation from OOA/ OOD models, CASE-tool support for all development process phases, Three level architecture (modelling, construction, and implementation level) [102] [104], Abstract and Concrete Interaction Objects, Code generation [103] [104] [105]
MASTERMIND	HTML [108], C++, Amulett Toolkit [110]	2D Graphical User Interfaces [107]	Open framework, All-encompassing design model, Continuity across the entire life cycle of the UI [106], Application wrapper, Model bindlings [108] [109]
JANUS	C++ [113] [114], HTML, Java (Swing) [114]	Microsoft Windows [113] [114], Motif[203], OS/2, Unix [113], Batch user interfaces [114], Databases: OB (object-oriented), Access, Oracle oder DB2 (relational) [114]	Dialog knowledge, Layout knowledge, Convertions, Self-descriptions [112]
FUSE	C++ [115] [117]	Not specified	Dialogue guidelines, Layout guidelines, Guideline Definition Layer, Guideline Application Layer, Dynamic On-Line Help [115]
MECANO	MECANO Runtiome Environment [118] [119]	Windows95 [119]	Automatic generation of static UI layout and dynamic UI behavior, High-level dialog, Low-level dialog, Execute textual interface-model instances by multiple run-time systems [118]
TADEUS (Rostock)	Existing UIMS, e.g. ISA Dialog Manager [120]	Windows-oriented GUI [120] [121]	Dialogue Graph [120] [121] [22], Navigation dialogue, Processing dialogue, View [120] [121] [21], Interaction table [120] [21]

[205] For details see e.g. [218]

MB-UIDE	Target Languages	Target Platforms	Major Concepts
MOBI-D	MOBI-D Runtime System, ActiveX Controls, Java applets [124]	Desktop PC [124] [125]	Interface Model [123] [124], Design Views [123], Executable interface specification [124]
TEALLACH	Java Bean and Swing [126]	UI for object-oriented database management systems (OODBMS) [126], further details not specified	Pre-existing domain application, Platform independence per Domain Model based on ODMG concepts, Task Model specifying task structure and information flow between models, Abstract and concrete Presentation Model, Associations between lements of the models [126]
TADEUS (LINZ)	Microsoft Foundation Class (MFC) Library [128]	Microsoft Windows, 2D and 3D platzforms [128]	Development approach combining model-driven, task-based, user-oriented, and object-driven techniques, TADEUS Task Analysis and Representation (TATAR), Incremental refinement of core models [128]
TERESA	X+V (graphical + vocal) SVG, Xlet, Microsoft gesture library [73], XFORMS (planned) [62], XHTML, VoiceXML [61] [73] [141]	Graphical desktop, Vocal, Cellphone, Graphical and vocal desktop, Graphical and gestural, digital TV [73]	High-level task model addressing possible contexts of use and target platforms, System task model per platform, Abstract UI, Concrete UI, Code generation [62]
SUPPLE (++)	Not specified	Mobile phone, Touch screen devices [129], Desktop computer [134]	User- and device-specific cost functions, Rendering based on solving decision-theoretic optimization problem, usage patterns [129], User traces [129] [134], Distributes Architecture [134]
MARIAE	Java [137], XHTML [137] [142]	Graphical from-based, Graphical mobile form-based, Vocal, Digital TV, Graphical direct manipulation, multimodal (graphical and vocal) for desktop and mobile, Advanced mobile (multitouch / accelerometers), XHTML + Safari DOM extension [65]	Successor of TERESA, Customizable model-to-model transformations [65], Implemented according to the CAMELEON Reference Framework (CRF), Web Service Annotations [136]

Tabelle 47 ERGEBNISSE MB-UIDE-VERGLEICH – TEIL 4

MB-UIDE	Tools	Availability	Application Examples
UIDE	DON [80] [85] [86], CARTOONIST [87] [80], USAGE, AutoHelp [80]	Existing implementation [81], executable version not located	Simple drawing program [79] [81], Digital curcuit design application [82] [84]
ITS	Dialog Compiler, Style Compiler [88]	Existing implementation [89] [90], executable version not located	Visitor Information System of EXPO '92 [89] [90], Doalog workbenches [90]
HUMANOID	Model editors [93], Runtime Environment [92] [93]	Existing implementation [92] [93] [94] [91], executable version not located	Agenda management tool, Inventory control system [91], TreeViz (part of HUMANOID) [93], Object browser [92] [94], Logistics analysis system (DRAMA), Knowledge base development environment (SHELTER), HUMANOID interactive design environment itself [91] [92] [93]
ADEPT	Task Model Editor, User Model Editor, AUI Editor, Interface Generator (Model to AUI), UI Generator (AUI to CUI) [95] [96]	Existing implementation [95] [96], executable version not located	CAD tool for jewellery designers [95], Airline flight query and booking system [97]
GENIUS	ER Diagram Editor (ERDE), Graphical Dialog Net Editor [98]	Existing implementation [98], executable version not located	Customer order system [98]
TRIDENT	Activity Chaining Graph (ACG) Editor [100], Specification Editor, AIO-to-CIO Mapper, CIO Placer, Presentation Editor [23]	Existing implementation [100], executable version not located	Tool supporting phone order process of a clothes selling company [100] [99]
AME	OODevelopTool [102] [103] [104], Structure Refinement Tool, Behavior Tool, Presentation Tool [102], CreateASTool [102] (no longer supported later on [103])	Existing implementation [102] [103] [104] [105], executable version not located	Language translation assistance for existing text processor (TRANTOOL) [103] [104]
MASTERMIND	Code Generators [108], Graphical dialogue-modeling tool (Dukas) [108], Dialogue parser / interpreter [109]	Existing implementation [80] [108], executable version not located	Electronic Mail Application [107], Web Browser [108] [109], Air Traffic Control Application [110]

MB-UIDE	Tools	Availability	Application Examples
JANUS	JANUS Layout Expert System [112], JANUS Application Framework (JAF), JANUS Application Development Framework (JADE), GUI Generator, Application Generator [113], other existing OO CASE tools [113]	Existing implementation [112] [113] [114], executable version not located	Seminar organisation application [112], Simple corporate employee manahgement system [113], Product and supplier management application [114]
FUSE	Bedien-Oberflächen-Spezifikations-System (BOSS), Formal User Interface Development (FLUID), Plan–based User Guidance for Intelligent Navigation (PLUG-IN), Formal Interface Requirements Engineering (FIRE) [115]	Existing implementation [115], executable version not located	ISDN phone simulation, Literature retrieval system, Home banking system, Formula editor for LATEX [115]
MECANO	Design tools: Model Editor, Intelligent Designer, Interface Builder [118] Runtime tool: Performance Monitor [118]	Existing implementation [118] [119], executable version not located	Medical therapy planning application, Elevator configuration application [118]
TADEUS (Rostock)	Dialog Graph Editor, Existing Petri Net tool for Dialog Graph simulation, Existing UIMS (ISA Dialog Manager) [120], User Modelling Component [120] [21]	Existing implementation [120], executable version not located	Information System GUI [120], TADEUS User Modeling Component [120] [21], Banking System [122]
MOBI-D	User-task Elicitation Tool (U-TEL), Interactive model editors (user-task, domain, design components), Interface Design Assistant [124], Task-based Interface Builder, Design Model Editor, The Interface Model Mapper (TIMM) [125]	Existing implementation [123] [124], executable version not located	Interactive logistics map [123], Military logistics operations application [124] [125], Ship protection system [119]

MB-UIDE	Tools	Availability	Application Examples
TEALLACH	Overall tool environment providing project management, editing, model linking, and code generation includeing Task Model Editor, Data Model Editor, Presentation Model Editor, and Java code genarator [126]	Existing implementation [126], executable version not located	Library management application [126]
TADEUS (LINZ)	Interactive Workspace, Consistency Checker, Prototyping Engine, Repository [128]	Existing implementation [128], executable version not located	Human ressource management application [128]
TERESA	ConcurTaskTrees Environment (CTTE), EditorFrame, Code generators [73]	Existing implementation [58] [62] [74] [73] [141], downloadable from the Internet[206]	Museum guide [62] [73], Diverse application examples [73]
SUPPLE (++)	ARNAULD [131] [132] [135], Activity Modeler [132] [135]	Existing implementation [129] [130] [132] [133] [134] [135], executable version not located	FTP client, Controller for classroom equipment [129], Email client, Interface to Amazon Web Services [134], Printer Setup Dialog [132] [134]
MARIAE	Transformation Editor [65] [137], Tasks-Services Binding Editor, User Interface Editor (AUI and CUI), FUI Preview [65]	Existing implementation [64] [65] [76] [136] [137] [138] [142], downloadable from the Internet[207]	Pac-Man game [65], Home control application [64], Sales order management [138], DVD management application [136]

[206] http://giove.cnuce.cnr.it/teresa.html
[207] http://giove.isti.cnr.it/tools/MARIAE/download

C. Übersichten zu den untersuchten Pattern-Sammlungen

C.1 Organisation der Pattern-Sammlung *Designing Interfaces*

Tabelle 48 ORGANISATION DER PATTERN-SAMMLUNG VON JENIFER TIDWELL

Kategorie	Anzahl	Patterns
What users do	14	Safe exploration; Instant gratification; Satisficing; Changes in midstream; Deferred choices; Incremental construction; Habituation; Microbreaks; Spatial memory; Prospective memory; Streamlined repetition; Keyboard only; Other people's advice; Personal recommendations
Organizing the content	10	Feature, search and browse; News stream; Picture manager; Dashboard; Canvas plus palette; Wizard; Settings editor; Alternative views; Many workspaces; Multi-level help
Getting around	13	Clear entry points; Menu page; Pyramid; Modal panel; Deep-linked state; Escape hatch; Fat menus; Sitemap footer; Sign-in tools; Sequence map; Breadcrumbs; Annotated scrollbar; Animated transition
Organizing the page	13	Visual framework; Center stage; Grid of equals; Titled sections; Module tabs; Accordion; Collapsible panels; Movable panels; Right/left alignment; Diagonal balance; Responsive disclosure; Responsive enabling; Liquid layout
Lists of things	12	Two-panel selector; One-window drilldown; List inlay; Thumbnail grid; Carousel; Row striping; Pagination; Jump to item; Alphabet scroller; Cascading lists; Tree table; New-item row
Doing things	11	Button groups; Hover tools; Action panel; Prominent "done" button; Smart menu items; Preview; Progress indicator; Cancelability; Multi-level undo; Command history; Macros
Showing complex data	11	Overview plus detail; Datatips; Data spotlight; Dynamic queries; Data brushing; Local zooming; Sortable table; Radial table; Multi-Y graph; Small multiples; Treemap
Getting input from users	11	Forgiving format; Structured format; Fill-in-the-blanks; Input hints; Input prompt; Password strength meter; Autocompletion; Dropdown chooser; List builder; Good defaults; Same-page error messages
Using social media	12	Editiorial mix; Personal voices; Repost and comment; Conversation starters; Inverted nano-pyramid; Timing strategy; Specialized streams; Social links; Sharing widget; News box; Content leaderboard; Recent chatter
Going mobile	11	Vertical stack; Filmstrip; Touch tools; Bottom navigation; Thumbnail-and-text list; Infinite list; Generous borders; Text clear button; Loading indicators; Richly connected apps; Streamlined branding
Make it look good	7	Deep background; Few hues, many values; Corner treatments; Borders that echo fonts; Hairlines; Contrasting font weights; Skins and themes

C.2 Organisation der Pattern-Sammlung *Patterns in Interaction Design*

Tabelle 49 ORGANISATION DER PATTERN-SAMMLUNG VON MARTIJN VAN WELIE HAUPTKATEGORIE "USER NEEDS"

Unterkategorie	Anzahl	Patterns
Navigating around	25	Accordion; Headerless menu; Breadcrumbs; Directory navigation; Doormat navigation; Double tab navigation; Faceted navigation; Fly-out menu; Home link; Icon menu; Main navigation; Map navigator; Meta navigation; Minesweeping; Panning navigator; Overlay menu; Repeated menu; Retractable menu; Scrolling menu; Shortcut box; Split navigation; Teaser menu; To-the-top link; Trail menu; Navigation tree
Basic interactions	7	Action button; Guided tour, Paging; Pulldown button; Slideshow; Stepping; Wizard
Searching	13	Advanced search; Autocomplete; Frequently asked questions (FAQ); Help wizard; Search box; Search area; Search results; Search tips; Site index; Site map; Footer sitemap; Tag cloud; Topic pages
Dealing with data	14	Carrousel; Table filter; Collapsible panels; Detail on demand; Collector; Inplace replacement; List builder; List entry view; Overview by detail; Parts selector; Tabs; Table sorter; Thumbnail; View
Personalizing	3	Customizable window; Login; Registration
Shopping	9	Booking; Product comparison; Product advisor; Product configurator; Purchase process; Shopping cart; Store locator; Testimonials; Virtual product display
Making choices	5	Country selector; Date selector; Language selector; Poll; Rating
Giving input	3	Comment box; Constraint input; Form
Miscelleaneous	5	Footer bar; Hotlist; News box; News ticker; Send-a-friend link

Tabelle 50 ORGANISATION DER PATTERN-SAMMLUNG VON MARTIJN VAN WELIE HAUPTKATEGORIE "APPLICATION NEEDS"

Unterkategorie	Anzahl	Patterns
Drawing attention	8	Captcha; Center stage; Color coded section; Premium content lock; Grid-based layout; Liquid layout; Outgoing links; Alternating row colors
Feedback	2	Input error message; Processing page
Simplyfying interaction	2	Enlarged clickarea; Font enlarger

Tabelle 51 ORGANISATION DER PATTERN-SAMMLUNG VON MARTIJN VAN WELIE HAUPTKATEGORIE "CONTEXT OF DESIGN"

Unterkategorie	Anzahl	Patterns
Site types	14	Web-based application; Artist site; Automotive site; Branded promotion site; Campaign site; E-commerce site; Community site; Corporate site; Multinational site; Museum site; Personalized 'my' site; News site; Portal site; Travel site
Experiences	8	Community building; Information management; Fun; Information seeking; Learning; Assistance; Shopping; Story telling
Page types	13	Article page; Blog page; Case study; Contact page; Event calendar; Forum; Guest book; Help page; Homepage; Newsletter; Printer-friendly page; Product page; Tutorial

C.3 Organisation der Pattern-Sammlung *Design of Sites*

Tabelle 52 ORGANISATION DER PATTERN-SAMMLUNG VON DOUGLAS VAN DUYNE

Kategorie	Anzahl	Patterns
Site genres	12	Personal e-commerce; News mosaics; Community conference; Self-service government; Nonprofits as networks of help; Grassroots information sites; Valuable company sites; Educational forums; Stimulating arts and entertainment; Web apps that work; Enabling intranets; Blogs
Creating a navigation framework	9	Multiple ways to navigate; Browsable content; Hierarchical organization; Task-based organization; Alphabetical organization; Chronological organization; Popularity-based organization; Category pages; Site accessibility
Creating a powerful homepage	2	Homepage portal; Up-front value proposition
Writing and managing concept	11	Page templates; Content modules; Headlines and blurbs; Personalized content; Message boards; Writing for search engines; Inverted-pyramid writing style; Printable pages; Distinctive HTML titles; Internationalized and localized content; Style sheets
Building trust and credibility	9	Site branding; E-mail subscriptions; Fair information practices; Privacy policy; About us; Secure connections; E-mail notifications; Privacy preferences; Preventing phishing scams
Basic e-commerce	9	Quick-flow checkout; Clean product details; Shopping cart; Quick address selection; Quick shipping method selection; Payment method; Order summary; Order confirmation and thank-you; Easy returns
Advanced e-commerce	7	Featured products; Cross-selling and up-selling; Personalized recommendations; Recommendation community; Multiple destinations; Gift giving; Order tracking and history
Helping customers to complete tasks	13	Process funnel; Sign-in/new account; Guest account; Account management; Persistent customer sessions; Floating windows; Frequently asked questions; Context-sensitive help; Direct manipulation; Clear forms; Predictive input; Drill-down options; Progress bar

Designing effective page layouts	6	Grid layout; Above the fold; Clear first reads; Expanding screen width; Fixed screen width; Consistent sidebars of related content
Making site search fast and relevant	3	Search action module; Straightforward search forms; Organized search results
Making navigation easy	17	Unified browsing hierarchy; Navigation bar; Tab rows; Action buttons; High-visibility action buttons; Location bread crumbs; Embedded links; External links; Descriptive, longer link names; Obvious links; Familiar language; Preventing errors; Meaningful error messages; Page not found; Permalinks; Jump menus; Site map
Speeding up your site	6	Low number of files; Fast-loading images; Separate tables; HTML power; Reusable images; Fast-loading content
The mobile web	3	Mobile screen sizing; Mobile input controls; Location-based services

C.4 Organisation der Pattern-Sammlung *Quince Pattern Library*

Tabelle 53 ORGANISATION DER QUINCE PATTERN LIBRARY, ZUORDNUNG VON PATTERNS ZU PATTERN TAGS

	Pattern Tag																																										
Pattern Name	Displaying Complex Data	Data Visualization	Drill-Down	Map	Spatio-Temporal	Analysis	Data Analysis	Chart	Grid	Tree	Hierarchy	Multiple Selection	Data Entry	Format	Rating	Usability	Commands	Visual Hierarchy	Date Entry	Help	Notifications	Exception Handling	Builders and Editors	Interaction Design	Navigation	Tagging	Search	Information Design	Information Architecture	Browse	Filtering	Selection	Form	Selector	Text Input	Masked Edit	Numeric Input	Page Layout	Alignment	Customization	Visual Design	Color	Constistency
Action Links																	x																											
Active Filtering	x	x				x																																					
Alphanumeric Filter Links																									x		x			x													
Alternating Row Colors	x								x							x																											
Alternative Views	x						x																						x										x				
Annotated Scroll Bar																									x		x			x													
Bread Crumbs																									x																			
Button Groups																	x															x											
Cascading Lists											x		x														x				x												
Clear Entry Points																x									x		x		x	x							x						
Closable Panels																								x	x				x									x					
Command Area																	x							x					x														
Corner Treatments																																									x	x	x
Dashboard	x	x				x		x	x																																		
Data Brushing	x						x	x																																			
Data Tips	x	x	x				x	x																																			
Data Visualization		x		x	x																																						
Date and Time Input													x						x																								
Date Picker													x						x													x											
Date Time Range Input													x						x													x											

Pattern Name	Displaying Complex Data	Data Visualization	Drill-Down	Map	Spatio-Temporal	Analysis	Data Analysis	Chart	Grid	Tree	Hierarchy	Multiple Selection	Data Entry	Format	Rating	Usability	Commands	Visual Hierarchy	Date Entry	Help	Notifications	Exception Handling	Builders and Editors	Interaction Design	Navigation	Tagging	Search	Information Design	Information Architecture	Browse	Filtering	Selection	Form	Selector	Text Input	Masked Edit	Numeric Input	Page Layout	Alignment	Customization	Visual Design	Color	Consistency
Drop Down Button																	x	x																									
Drop Down Chooser													x																				x	x									
Edit-in-Place													x										x																				
Extras on Demand																								x				x	x				x										
Faceted Navigation						x																			x		x		x	x													
Few Hues																																									x		
Forgiving Format													x	x					x																								
Form													x																				x										
Formatted Text Input													x																				x		x								
Global Navigation										x	x														x																		
Grid Layout																																						x					
Hub and Spoke																									x				x														
Illustrated Choices													x												x																x		
Inline Validation													x								x	x																					
Input Hints													x							x																							
Input Prompt													x							x																							
Intriguing Branches																									x				x	x													
Invitation																								x	x					x													
Journal Navigation																									x																		
Large Set Single Selector									x	x			x																														
Left Aligned Labels																																	x					x	x				
Liquid Layout																								x														x		x			
List Sorter													x										x	x									x										

(The column headings are printed under the spanning heading **Pattern Tag**.)

Pattern Name	Pattern Tag																																										
	Displaying Complex Data	Data Visualization	Drill-Down	Map	Spatio-Temporal	Analysis	Data Analysis	Chart	Grid	Tree	Hierarchy	Multiple Selection	Data Entry	Format	Rating	Usability	Commands	Visual Hierarchy	Date Entry	Help	Notifications	Exception Handling	Builders and Editors	Interaction Design	Navigation	Tagging	Search	Information Design	Information Architecture	Browse	Filtering	Selection	Form	Selector	Text Input	Masked Edit	Numeric Input	Page Layout	Alignment	Customization	Visual Design	Color	Consistency
Local Zooming	x			x					x																																		
Magnetism																x							x	x														x	x	x			
Map	x			x																					x			x															
Modal Panel													x			x		x					x	x	x																		
Movable Panels																																						x		x			
Multiple Selection from a Large List									x	x		x	x																														
Multiple Selection from a Small List									x	x		x	x																														
Multi-Y Graph	x							x																																			
Navigation Tabs																									x																		
New-Item Row									x				x											x									x										
Number in Range Input													x		x																		x				x						
One-Window Drilldown			x																										x														
Overview Plus Detail	x			x																																							
Paging																									x																		
Plain Text Input													x																				x		x								
Preview			x											x	x		x						x		x					x		x						x	x	x			
Primary Action													x			x	x	x															x					x	x				
Progress Indicator													x			x	x										x						x										
Property Sheet													x										x															x					
Real Time Monitor	x																																										
Responsive Disclosure																								x									x					x					
Responsive Enabling																																	x					x					
Ribbon																	x								x																		

Pattern Name	Pattern Tag																																										
	Displaying Complex Data	Data Visualization	Drill-Down	Map	Spatio-Temporal	Analysis	Data Analysis	Chart	Grid	Tree	Hierarchy	Multiple Selection	Data Entry	Format	Rating	Usability	Commands	Visual Hierarchy	Date Entry	Help	Notifications	Exception Handling	Builders and Editors	Interaction Design	Navigation	Tagging	Search	Information Design	Information Architecture	Browse	Filtering	Selection	Form	Selector	Text Input	Masked Edit	Numeric Input	Page Layout	Alignment	Customization	Visual Design	Color	Consistency
Right Aligned Labels																																	x					x	x				
Same Page Error Messages													x																				x										
Search																									x		x																
Search Results																									x		x																
Skins																																									x		
Small Multiples	x																																										
Small Set Single Selector													x																				x										
Smart Menu Items																	x																										
Sortable Table	x								x																																		
Status Area																	x																					x					
Structured Format													x																				x			x							
Tab Dialogs																																						x					
Table Filter	x					x			x																		x																
Tag Cloud	x					x																			x	x																	
Task Pane																	x																										
Text Field Autocompletion													x																														
Tiled Sections																		x										x	x									x			x		
Top Aligned Labels																																	x					x	x				
Transition																x								x	x																		
Tree Map	x																																										
Tree-Table	x									x	x																																
Two-Panel Selector																									x					x	x	x						x					
Undo													x			x	x							x																			

	Pattern Tag																																										
Pattern Name	Displaying Complex Data	Data Visualization	Drill-Down	Map	Spatio-Temporal	Analysis	Data Analysis	Chart	Grid	Tree	Hierarchy	Multiple Selection	Data Entry	Format	Rating	Usability	Commands	Visual Hierarchy	Date Entry	Help	Notifications	Exception Handling	Builders and Editors	Interaction Design	Navigation	Tagging	Search	Information Design	Information Architecture	Browse	Filtering	Selection	Form	Selector	Text Input	Masked Edit	Numeric Input	Page Layout	Alignment	Customization	Visual Design	Color	Consistency
Visual Framework																x								x	x			x										x			x	x	x
Wizard													x			x								x																			
Work With									x	x		x	x												x																		

Tabelle 54 ORGANISATION DER QUINCE PATTERN LIBRARY
ZUORDNUNG VON PATTERNS ZU USER TASKS

Pattern Name	User Task			
	Edit Things	Analyze Data	Find Stuff	Explore Info
Displaying Complex Data		x		
Data Visualization		x		
Drill-Down		x		x
Map		x		
Data Analysis		x		
Chart		x		
Tree				x
Hierarchy			x	x
Multiple Selection	x			
Data Entry	x			
Rating	x			
Builders and Editors	x			x
Navigation			x	
Tagging			x	
Search			x	
Information Architecture			x	
Browse			x	
Filtering		x		x
Form	x			
Text Input	x			
Masked Edit	x			
Numeric Input	x			
Alignment	x			

C.5 Organisation der Pattern-Sammlung *Yahoo Design Pattern Library*

Tabelle 55 ORGANISATION DER YAHOO DESIGN PATTERN LIBRARY

Hauptkategorie	Unterkategorie	Anzahl	Patterns
Layout	-	1	Page Grids
Navigation	-	3	Accordion, Alphanumeric Filter Links, Breadcrumbs
	Pagination	2	Item Pagination, Search Pagination
	Tabs	2	Module Tabs, Navigation Tabs
	Navigation Bar	3	Top Navigation Bar, Left ~, Progress Bar
Selection	-	3	Autocomplete, Calendar Picker, Corousel
Rich Interaction	Invitation	4	Cursor Invitation, Drop ~, Hover ~, Tooltip ~
	Transition	11	Animate Transition, Brighten ~, Collapse ~, Cross Fade ~, Dim ~, Expand ~,Fade In ~, Fade Out ~, Self Healing ~, Slide ~, Spotlight ~
	Drag and Drop	1	Drag and Drop Modules
Social	Core Principles	2	Talk Like a Person, Your vs. My
	People	16	Siehe Tabelle 56
	Objects	7	Siehe Tabelle 57

Tabelle 56 ORGANISATION DER YAHOO DESIGN PATTERN LIBRARY
HAUPTKATEGORIE "SOCIAL", UNTERKATEGORIE "PEOPLE"

Unter-Unterkategorie	Anzahl	Patterns
Engagement	4	Invite, Yor're Invited, Sign-in Continuity, Terms of Service
Identity	2	Reflector, User Card
Presence	2	Availability, Updates
Reputation	6	The Competitive Spectrum, Collectible Achievements, Identifying Labels, Named Labels, Numbered Levels, Points
Ranking	2	Leaderboard, Top X

Tabelle 57 ORGANISATION DER YAHOO DESIGN PATTERN LIBRARY
HAUPTKATEGORIE "SOCIAL", UNTERKATEGORIE "OBJECTS"

Unter-Unterkategorie	Anzahl	Patterns
Collecting	3	Add/Subscribe, Favorites, Saving
Feedback	4	Architecture of Review, Rating an Object, Vote to Promote, Write a Review

C.6 Mapping der Pattern-Beschreibungen auf PLML 1.1

Tabelle 58 VERGLEICH DER IN DEN UNTERSUCHTEN PATTERN-SAMMLUNGEN VERWENDETEN BESCHREIBUNGSELEMENTE MIT PLML 1.1

PLML 1.1	Designing Interfaces	Patterns in Interaction Design	Design of Sites	Quince Pattern Library	Yahoo Design Pattern Library
<patternID>	-	-	Kategorie und fortlaufende Nummer	-	-
<name>	<name>	<name>	<name>	<name>	<name>
<alias>	-	-	-	-	-
<illustration>	<figure>	enthalten in <solution>	<figure>	-	<illustration>
<problem>	enthalten in <use-when>	<problem>	<problem>	<problem>	<problem>
<context>	enthalten in <use-when>	<use-when>	teilweise enthalten in <background> und teilweise in <problem>	<context>	<context>
<forces>	-	-	-	-	-
<solution>	<what> und <how>	<solution> und <how>	<solution>	<solution>	<solution>
<synopsis>	-	-	-	-	-
<diagram>	-	gelegentlich enthalten in <how>	enthalten in <solution>	-	gelegentlich enthalten in <illustration>
<evidence> <example> <rationale>	<examples> <why>	<more-examples> <why>	- enthalten in <problem>	<examples> <rationale>	<examples>, <as-used-on-Yahoo>, <other-examples> und <pattern-gallery> <rationale>
<confidence>	-	-	-	-	-
<literature>	gelegentlich enthalten in <in-other-libraries>	<literature>	-	<sources>	-

PLML 1.1	Designing Interfaces	Patterns in Interaction Design	Design of Sites	Quince Pattern Library	Yahoo Design Pattern Library
<implementation>	-	-	gelegentlich enthalten in <problem>	<implementation>	<accessibility>, <code-examples>, <special-cases> und gelegentlich auch enthalten in <solution>
<related-patterns>	gelegentlich enthalten in <why> oder <how>	gelegentlich enthalten in <use-when>, <how> oder <why>	<other-patterns-to-consider> und gelegentlich enthalten in <problem> oder <background>	Kategorisierung mittels <tags>	<related-patterns>, <similar-patterns-in-other-libraries> und <disambiguation>
<pattern-link> <type> <patternID> <collection> <label>	-	-	- Kategorie und fortlaufende Nummer - -	-	-
<management> <author> <credits> <creation-date> <last-modified> <revision-number>	-	-	-	-	- - - <last-modified> -
-	-	<comments>	-	<comments>	<blogs>
-	-	-	-	<tags>	-
-	-	-	-	-	<status>
-	-	-	-	-	<open-questions>
-	-	-	-	-	<wire-frame-stencils>

D. Vergleich von Pattern-Werkzeugen im Detail

Tabelle 59 ERGEBNISSE PATTERN-TOOL-VERGLEICH – TEIL 1

Pattern Tool	Originator	Start of Dvpt. / First Publ.	Current Version / Latest Publ.	Availability
UPADE	Concordia University, Montreal [188] [189]	2001 [188]	2003 [189]	Existing Java prototype [189], executable version not located
Damask	Design: University of California, Berkeley [190], Implementation: IBM Almaden Research Center, San José and University of Washington, Seattle [191]	2002 [190]	2008 [191]	Existing implementation [191], executable version not located
MOUDIL	Concordia University, Montreal [192] [193] [194]	2003 [192]	2005 [193] [194]	Existing implementation [193], executable version not located
MUIP	Massey University, Palmerston North [187] [174]	2005 [187]	2006 [174]	Existing implementation [187], executable version not located
PLAss	Fachhochschule Augsburg [195]	2006 [195]	2006 [195]	Existing implementation [195], available at Hochschule Augsburg
HCI PLMT	American University in Cairo [196]	2006 [196]	2006 [196]	Existing implementation [196], executable version not located
PIM Tool	University of Rostock and Concordia University, Montreal [20]	2007 [20]	2007 [20]	Existing implementation [20], executable version not located
Task Pattern Wizard	Concordia University, Montreal [197]	2007 [197]	2007 [197]	Existing implementation [197], executable version not located
PatternWiki	RWTH Aachen and Universität Duisburg-Essen [198]	2014 [198]	2014 [198]	Existing implementation [198], accessible via Internet
Website *Designing Interfaces*	Jenifer Tidwell [199] [182]	Not identified	Last website update in 2011 [182]	Existing implementation [182], accessible via Internet

Pattern Tool	Originator	Start of Dvpt. / First Publ.	Current Version / Latest Publ.	Availability
Website *Patterns in Interaction Design*	Martijn van Welie [183]	Not identified	Last website update in 2008 [183]	Existing implementation [183], accessible via Internet
Website *Design of Sites*	Douglas van Duyne [184]	Not identified	Last website update in 2009 [184]	Existing implementation [184], accessible via Internet
Yahoo Design Pattern Library	Yahoo! Inc. [185]	Not identified	Not identified	Existing implementation [185], accessible via Internet
Quince UX Patterns Explorer	Infragistics Inc. [186]	Not identified	Not identified	Existing implementation [186], accessible via Internet

Tabelle 60 ERGEBNISSE PATTERN-TOOL-VERGLEICH – TEIL 2

Pattern Tool	Used Pattern Description Language	Used Pattern Description Elements	MajorConcepts	Supported Pattern Application Features
UPADE	PCML [189]	See chapter 3.2.1.5	Three different abstraction levels: pattern, design and code level [189]	Pattern composition [188] [189], code generation [189]
Damask	PLML 1.1 [191]	See chapter 3.2.1.1	Sketch-based cross-device UI design, prebuilt device-specific UI design fragments, which can be manually changed after being applied, created UI designs can be executed by device simulators [190] [191], layer approach to distinguish general and device-specific parts [191]	Integration of device-specific UI design fragments into UI designs [190] [191]

Pattern Tool	Used Pattern Description Language	Used Pattern Description Elements	MajorConcepts	Supported Pattern Application Features
MOUDIL	Proprietary [193] (not fully described)	<Head>, <Body>, <Relations>, <Assimilation> [193]	Seven C-s methodology [192] [193] [194], separated pattern author and user roles [193] [194], international editorial board to review, validate, and acknowledge patterns [192] [193]	Automated context-sensitive advice for MOUDIL users, generation of Java or C# classes [193]
MUIP	PLML 1.2 [174] [187]	See chapter 3.2.1.2	<forces> can be manipulated by users and support browsing, searching, and categorizing of patterns [174] [187]	None [174] [187]
PLAss	Any [195]	Freely configurable [195]	Separation of author and user roles, automated support for pattern relation specification [195]	None [195]
HCI PLMT	Proprietary [196] (not fully described)	Mandatory: <Name>, <Problem>, <Solution>, <Web-Site-Link>, <Category> Optional: <Context> [196]	Ontological approach, alternative and related patterns are identified on the basis of string matching operations [196]	None [196]
PIM Tool	TPML [20]	See chapter 3.2.1.6	Patterns contain model fragments, combination of model-driven and pattern-based approaches [20]	Pattern identification, selection, instantiation, and integration into UI multi-model [20]
Task Pattern Wizard	Proprietary [197] (not fully described)	<Problem>, <Context>, <Solution>, <Rationale> [197]	Patterns contain model fragments (XUL), combination of model-driven and pattern-based approaches [197]	Pattern identification, selection, instantiation, and integration into task, dialog, presentation, and layout models [197]
PatternWiki	PLML (version not specified) [198]	See chapters 3.2.1.1 and 3.2.1.2	MediaWiki engine with extensions for XML representations and graph-based visualization [198]	XSLT-based transformations [198]

Pattern Tool	Used Pattern Description Language	Used Pattern Description Elements	MajorConcepts	Supported Pattern Application Features
Website *Designing Interfaces*	Proprietary [182]	See chapter 3.3.2.1	Navigation panel shows available patterns per category [199]	None [182] [199]
Website *Patterns in Interaction Design*	Proprietary [183]	See chapter 3.3.2.2	Links to other pattern compilations, user may suggest new patterns and send messages to the author [183]	None [183]
Website *Design of Sites*	Proprietary [184]	See chapter 3.3.2.3	Background information on patterns, user may post pattern application examples [184]	None [184]
Yahoo Design Pattern Library	Proprietary [185]	See chapter 3.3.2.4	Background information on patterns, links to other pattern compilations, list of recently updated pattern descriptions, downloadable desgin stencils in different formats [185]	None [185]
Quince UX Patterns Explorer	Proprietary [186]	See chapter 3.3.2.5	Background information on patterns, three different tool versions (public, registered users, professional), exploration of patterns by tasks, tags, and typical UI placement [186]	Design boards: Quince users may create flows that describe user behaviors and explain software paths (Quince Pro) [186]

Tabelle 61 ERGEBNISSE PATTERN-TOOL-VERGLEICH – TEIL 3

	Supported Pattern Management Features							Supported Collaboration Features			Tool Category		
Pattern Tool	**Authoring**	**Browsing**	**Searching**	**Modification**	**Relating**	**Manipulating Compilations / Private Workspace**	**Import / Export**	**Discussion**	**Review / Comment**	**Revision / Change Log**	**Pattern Catalogue Tool**	**Pattern Management Tool**	**Pattern-based Design Tool**
UPADE	X [188] [189]	X [188] [189]	X [189]	X [189]	X [189]	-	-	-	-	-	-	-	X
Damask	X [190]	X [190] [191]	X [191]	-	-	-	-	-	-	-	-	-	X
MOUDIL	X [192] [193]	X [192] [193]	X [192] [193]	X [192] [193]	X [192] [193]	-	X [192] [193]	-	X [192] [193]	-	-	X[208]	X[209]
MUIP	X [187] [174]	X [187] [174]	X [187] [174]	X [187] [174]	X [187] [174]	X [187] [174]	-	-	X [187] [174]	-	-	X	-
PLAss	X[210] [195]	X [195]	X [195]	X[210] [195]	X[210] [195]	-	X [195]	-	-	-	-	X	-
HCI PLMT	-	X [196]	X [196]	X [196]	X [196]	-	-	-	-	-	-	X	-
PIM Tool	-	-	-	-	-	-	-	-	-	-	-	-	X
Task Pattern Wizard	-	X [197]	-	-	-	-	-	-	-	-	-	-	X
PatternWiki	X [198]	X [198]	X [198]	X [198]	X [198]	-	-	X [198]	X [198]	X [198]	-	-	X
Website *Designing Interfaces*	-	X [182] [199]	X [182]	-	-	-	-	-	-	-	X		
Website *Patterns in Interaction Design*	-	X [183]	X [183]	-	-	-	-	-	X [183]	-	X	-	-
Website *Design of Sites*	-	X [184]	X [184]	-	-	-	-	-	X [184]	-	X	-	-
Yahoo Design Pattern Library	-	X [185]	X [185]	-	-	-	-	X[211] [185]	-	-	X	-	-
Quince UX Patterns Explorer	-	X[212] [186]	X[212] [186]	-	-	X[215] [186]	-	X[214] [186]	X[214] [186]	-	X[213]	X[214]	X[215]

[208] According to information provided in [192]
[209] According to information provided in [193]
[210] In open mode
[211] Via Yahoo Developer Network (YDN) blogs and forums
[212] Provided that *Microsoft Silverlight* is installed and activated (see http://www.microsoft.com/silverlight/)
[213] Public Version (user not registered or not signed-in)
[214] Public Version (user registered and signed-in)
[215] Quince Pro

E. Beschreibung der Modelle im Detail

In den folgenden Unterkapiteln werden die Beschreibungselemente der Aufgaben-, Konzept- und Dialog-Modelle im Detail beschrieben. Es ist jeweils der Name des Beschreibungselements, eine kurze Beschreibung, der Datentyp und die Kardinalittät angegeben.

Die Festlegung der Datentypen erfolgt gemäß folgender Konventionen:

- EMPTY Das Element ist immer leer und dienst nur zu Strukturierungszwecken
- #PCDATA Das Element enthält ausschließlich Text
- ANY Das Element kann beliebigen Inhalt haben
- Enumeration Das Element kann einen der in der betreffenden Fußnote spezifizierten Werte annehmen

Die Spezifikation der Kardinalität erfolgt gemäß folgender Konventionen:

- 1 Das Element ist verpflichtend und existiert genau einmal
- ? Das Element ist optional und existiert nicht oder genau einmal
- * Das Element tritt keinmal, einmal oder beliebig häufig auf
- + Das Element tritt mindestens einmal oder beliebig häufig auf

Teilweise werden zwei Angaben zur Kardinalität gemacht, wobei die zweite von runden Klammern umschlossen ist, z. B. „? (1)“. Hierbei spezifiziert die erste Angabe die generelle Kardinalität und der in Klammern angegebene Wert die Kardinalität in Abhängigkeit der Existenz des direkt übergeordneten Elements. Das genannte Beispiel ist demnach wie folgt zu lesen:

- ? Das zugehörige Element tritt generell keinmal oder genau einmal auf
- (1) Falls das übergeordnete Element existiert, also die Kardinalität „1“ besitzt, dann muss auch dieses Element exisitieren

E.1 Beschreibungselemente von PaMGIS-Modellen

Tabelle 62 BESCHREIBUNGSELEMENTE VON PAMGIS-MODELLEN <PAMGIS_MODEL>

Element	Beschreibung	Typ	Kardinalität
<UniqueModelID>	Eindeutiger Bezeichner des Modells	EMPTY	1
<ModelID>	Bezeichner des Modells	#PCDATA	1
<ModelVersion>	Versionsnummer des Modells, i.d.R. zweistellig, wird nach signifikanten Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<ModelRevision>	Revisionsnummer des Modells, i.d.R. zweistellig, wird nach kleineren Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<ModelType>	Typ des Modells, z. B. Domänen-Modell	Enumeration[216]	1
<ModelName>	Bezeichnung des Modells	#PCDATA	1
<ModelDescription>	Textuelle Beschreibung des Modells	#PCDATA	?
<ModelAnnotation>	Anmerkungen und Hinweise in Textform	#PCDATA	?
<ModelReferences>	Liste mit Referenzen (Web-Links)	EMPTY	?
<ModelReference>	Angaben zu einer Referenz	EMPTY	* (+)
<MOREF_LinkDisplayName>	Benutzerfreundliche Bezeichnung des Links	#PCDATA	?
<MOREF_URL>	Web-Adresse	#PCDATA	? (1)
<ModelUserComments>	Liste von Benutzerkommentaren	EMPTY	?
<ModelUserComment>	Kommentar eines Benutzers	EMPTY	* (+)
<MODUC_Author>	Name des Erstellers	#PCDATA	? (1)
<MODUC_Date>	Datum	#PCDATA	? (1)
<MODUC_Comment>	Kommentar des Benutzers	#PCDATA	? (1)
<DomainModelUsabilityFeedback>	Ergebnisse von Usability-Aktivitäten, z. B. User-Testing	ANY	?
<ModelSubjectMatter>	Eigentlicher Inhalt des Modells je nach Modelltyp	EMPTY	1

[216] (Domain Model | Task Model | Concept Model | Context Model | Device Model | Toolkit Model | User Model | Environment Model | AUI Model | CUI Model | FUI Model)

E.2 Beschreibungselemente eines Domänen-Modells

Tabelle 63 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES DOMÄNEN-MODELLS <DOMAINMODELREFERENCES>

Element	Beschreibung	Typ	Kardinalität
<DomainModelReferences>	Verweise auf die beiden Teilmodelle	EMPTY	1
<TaskModelReference	Verweis auf das zugehörige Aufgaben-Modell	EMPTY	1
<TSKREF_ModelID>	Eindeutiger Bezeichner des Aufgaben-Modells	#PCDATA	1
<TSKREF_ModelVersion>	Versionsnummer des Aufgaben-Modells	#PCDATA	1
<TSKREF_ModelRevision>	Revisionsnummer des Aufgaben-Modells	#PCDATA	1
<TSKREF_ModelName>	Bezeichnung des Aufgaben-Modells	#PCDATA	?
<ConceptModelReference>	Verweis auf das zugehörige Konzept-Modell	EMPTY	1
<CPTREF_ModelID>	Eindeutiger Bezeichner des Konzept-Modells	#PCDATA	1
<CPTREF_ModelVersion>	Versionsnummer des Konzept-Modells	#PCDATA	1
<CPTREF_ModelRevision>	Revisionsnummer des Konzept-Modells	#PCDATA	1
<CPTREF_ModelName>	Bezeichnung des Konzept-Modells	#PCDATA	?

E.2.1 Beschreibungselemente eines Aufgaben-Modells

Tabelle 64 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONTEXT-MODELLS <TASK>

Element	Beschreibung	Typ	Kardinalität
<Task>	Angaben zu einer Aufgabe	EMPTY	1
<TaskID>	Eindeutiger Bezeichner der Aufgabe	#PCDATA	1
<TaskIDOriginal>	Ursprünglicher Bezeichner der Aufgabe, falls dieser z. B. bei der Anwendung eines Patterns geändert wurde	#PCDATA	?
<TaskName>	Bezeichnung der Aufgabe	#PCDATA	1

Element	Beschreibung	Typ	Kardinalität
<TaskDescription>	Textuelle Beschreibung der Aufgabe	#PCDATA	?
<TaskType>	Art der Aufgabe (abstrakte, Benutzer-, Interaktions- oder Anwendungs-Aufgabe)	Enumeration[217]	1
<TaskOrigin>	Herkunft der Aufgabe, d.h. Pattern, über das sie ins Aufgaben-Modell gelangte	EMPTY	?
<TaskOriginPatternID>	Eindeutiger Bezeichner des Musters	#PCDATA	? (1)
<TaskOriginPatternVersion>	Versionsnummer des Musters	#PCDATA	? (1)
<TaskOriginPatternRevision>	Revisionsnummer des Musters	#PCDATA	?
<TaskOriginPatternName>	Bezeichnung des Musters	#PCDATA	? (1)
<ContextsOfUse>	Liste von Benutzungskontexten, in denen die Aufgabe vorkommt	EMPTY	?
<ContextOfUse>	Bestimmung eines Benutzungskontext-Modells	EMPTY	* (+)
<ContextID>	Eindeutige Bezeichnung des Kontext-Modells	#PCDATA	? (1)
<ContextVersion>	Versionsnummer des Kontext-Modells	#PCDATA	? (1)
<ContextRevision>	Revisionsnummer des Kontext-Modells	#PCDATA	? (1)
<ContextName>	Bezeichnung des Kontexts-Modells	#PCDATA	?
<ContextDescription>	Textuelle Beschreibung des Kontexts	#PCDATA	?
<Optional>	Kennzeichnung, ob die Ausführung der Aufgabe optional ist	Enumeration[218]	1
<Iterative>	Kennzeichnung, ob die Aufgabe mehrfach ausgeführt werden kann	Enumeration[218]	1
<Conditional>	Kennzeichnung, ob eine Bedingung hinsichtlich der Ausführung der Aufgabe vorliegt	Enumeration[218]	1
<Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor die Aufgabe ausgeführt werden kann	EMPTY	?
<Precondition>	Einzelne Vorbedingung	EMPTY[219]	* (+)
<Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem die Aufgabe ausgeführt wurde	EMPTY	?
<Postcondition>	Einzelne Nachbedingung	EMPTY[219]	* (+)
<Position>	Position der Aufgabe im Aufgaben-Modell	EMPTY	1
<Parent>	Spezifikation der direkt übergeordneten Aufgabe	EMPTY	1
<ParentID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ beim Wurzel-Element	#PCDATA	1
<ParentName>	Bezeichnung dieser Aufgabe	#PCDATA	?

[217] (Abstraction | User | Interaction | Application)
[218] (True | False)
[219] Weitergehende Beschreibungsstruktur siehe Tabelle 65

Element	Beschreibung	Typ	Kardinalität
<SiblingLeft>	Spezifikation der vorhergehenden Aufgabe	EMPTY	1
<SiblingLeftID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ falls nicht vorhanden	#PCDATA	1
<SiblingLeftName>	Bezeichnung dieser Aufgabe	#PCDATA	?
<SiblingRight>	Spezifikation der nachfolgenden Aufgabe	EMPTY	1
<SiblingRightID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ falls nicht vorhanden	#PCDATA	1
<SiblingRightName>	Bezeichnung dieser Aufgabe	#PCDATA	?
<TemporalOperator>	Zeitliche Beziehung zur nachfolgenden Aufgabe	Enumeration[220]	?
<UIConcepts>	Liste der UI-Konzepte, die für die Ausführung der Aufgabe benötigt werden	EMPTY	?
<UIConcept>	Spezifikation eines UI-Konzepts	EMPTY	* (+)
<UIConceptID>	Eindeutiger Bezeichner des UI-Konzepts	#PCDATA	? (1)
<UIConceptName>	Bezeichnung des UI-Konzepts	#PCDATA	?
<SubTasks>	Liste von Unteraufgaben	EMPTY	?
<SubTask>	Angaben zu einer Unteraufgabe	<Task>	* (+)

Tabelle 65 BESCHREIBUNGSELEMENTE VON VOR- UND NACHBEDINGUNGEN

Element	Beschreibung	Typ	Kardinalität
<Operator>	Operator, z. B. „equal“, „less“, „greater“, „less-or-equal“, „greater-or-equal“, „unequal“	#PCDATA	? (1)
<Operands>	Operanden des logischen Ausdrucks	EMPTY	? (1)
<Operand>	Einzelner Operand	EMPTY	* (+)
<OperandType>	Art des Operanden, z. B. „Concept“ oder „Constant“	#PCDATA	? (1)
<OperandID>	Eindeutiger Bezeichner des Operanden	#PCDATA	?
<OperandName>	Bezeichnung des Operanden	#PCDATA	?
<OperandElement>	Bezeichnung des relevanten Beschreibungselements	#PCDATA	?
<OperandValue>	Wert des Operanden	#PCDATA	?

[220] (SequentialEnabling | Disabling | Choice | Interleaving | Synchronization | SuspendResume | SequentialEnablingInfo)

E.2.2 Beschreibungselemente eines Konzept-Modells

Tabelle 66 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONZEPT-MODELLS <CONCEPTS>

Element	Beschreibung	Typ	Kardinalität
<Concepts>	Liste der im Modell enthaltenen Konzepte	EMPTY	1
<Concept>	Angaben zu einem Konzept	EMPTY	+
<CCPT_ConceptID>	Eindeutiger Bezeichner des Konzepts	#PCDATA	1
<CCPT_ConceptIDOriginal>	Ursprünglicher Bezeichner der Aufgabe, falls dieser z. B. bei der Anwendung eines Patterns geändert wurde	#PCDATA	?
<CCPT_ConceptName>	Bezeichnung des Konzepts	#PCDATA	1
<CCPT_Description>	Textuelle Beschreibung des Konzepts	#PCDATA	?
<CCPT_Label>	Bezeichnung des Konzepts, die in der Benutzeroberfläche verwendet wird	#PCDATA	?
<CCPT_Perceptible>	Festlegung, ob das Konzept durch den Benutzer wahrnehmbar (z. B. sichtbar) ist oder nicht	Enumeration[221]	1
<CCPT_Enabled>	Festlegung, ob das Konzept aktuell verwendet werden kann oder nicht	Enumeration[221]	1
<CCPT_Required>	Festlegung, ob das Konzept vom Benutzer verwendet werden muss oder nicht	Enumeration[221]	1
<CCPT_ConceptType>	Art des Konzepts, z. B. DataInput, DataOutput oder DataEdit	Enumeration[222]	1
<CCPT_DataType>	Datentyp des Konzepts	Enumeration[223]	?
<CCPT_Value>	Wert	#PCDATA	? (1)
<CCPT_ConceptOrigin>	Herkunft des Konzepts, d.h. Pattern, über das es ins Aufgaben-Modell gelangte	EMPTY	?
<CCPT_OriginPatternID>	Eindeutiger Bezeichner des Musters	#PCDATA	? (1)
<CCPT_OriginPatternVersion>	Versionsnummer des Musters	#PCDATA	? (1)
<CCPT_OriginPatternRevision>	Revisionsnummer des Musters	#PCDATA	? (1)
<CCPT_OriginPatternName>	Bezeichnung des Musters	#PCDATA	?
<CCPT_Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor das Konzept verwendet werden kann	EMPTY	?

[221] (True | False)
[222] Siehe Tabelle 22
[223] (Integer | Float | String | Boolean | Picture)

Element	Beschreibung	Typ	Kardinalität
<CCPT_Precondition>	Einzelne Vorbedingung	EMPTY[224]	*
<CCPT_Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem das Konzept verwendet wurde	EMPTY	?
<CCPT_Postcondition>	Einzelne Nachbedingung	EMPTY[224]	*
<CCPT_DataLink>	Verweis auf ein Datenelement im Datenmodell der zugrunde liegenden Anwendung	EMPTY	?
<CCPT_DataLinkID>	Eindeutiger Bezeichner des Verweises	#PCDATA	? (1)
<CCPT_DataLinkName>	Bezeichnung des Verweises	#PCDATA	? (1)
<CCPT_DataLinkDescription>	Texuelle Beschreibung des Verweises	#PCDATA	?
<CCPT_DataLinkReference>	Referenz zum betreffenden Datenelement, z. B. der Name einer Klasse	#PCDATA	? (1)
<CCPT_TaskLinks>	Liste von Verweisen auf Aufgaben im Aufgaben-Modell, die das Konzept benötigen	EMPTY	?
<CCPT_TaskLink>	Einzelner Verweis auf eine Aufgabe	EMPTY	*
<CCPT_TaskID>	Eindeutiger Bezeichner dieser Aufgabe	#PCDATA	1
<CCPT_TaskName>	Bezeichnung dieser Aufgabe	#PCDATA	1

E.3 Beschreibungselemente eines Dialog-Modells

Tabelle 67 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES DIALOG-MODELLS <DIALOGCHARACTERISTICS>

Element	Beschreibung	Typ	Kardinalität
<DialogCharacteristics>	Eigenschaften der Dialoge	EMPTY	1
<DomainModelReference>	Verweis auf das zugehörige Domänen-Modell	EMPTY	?
<DOMREF_ModelID>	Eindeutiger Bezeichner des Kontext-Modells	#PCDATA	? (1)
<DOMREF_ModelVersion>	Versionsnummer des Domänen-Modells	#PCDATA	? (1)
<DOMREF_ModelRevision>	Revisionsnummer des Domänen-Modells	#PCDATA	? (1)

[224] Weitergehende Beschreibungsstruktur siehe Tabelle 65

Element	Beschreibung	Typ	Kardinalität
<DOMREF_ModelName>	Bezeichnung des Domänen-Modells	#PCDATA	?
<ContextModelReferences>	Liste von referenzierten Kontext-Modellen, für die das Dialog-Modell erstellt wurde	EMPTY	?
<ContextModelReference>	Einzelnes referenziertes Kontext-Modell	EMPTY	* (+)
<CTXREF_ModelID>	Eindeutiger Bezeichner des Kontext-Modells	#PCDATA	? (1)
<CTXREF_ModelVersion>	Versionsnummer des Kontext-Modells	#PCDATA	? (1)
<CTXREF_ModelRevision>	Revisionsnummer des Kontext-Modells	#PCDATA	? (1)
<CTXREF_ModelName>	Bezeichnung des Kontext-Modells	#PCDATA	?
<Dialogs>	Liste der im Modell enthaltenen Dialoge	EMPTY	1
<Dialog>	Angaben zu einem Dialog	EMPTY	+
<DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	1
<DialogName>	Bezeichnung des Dialogs	#PCDATA	1
<DialogDescription>	Textuelle Beschreibung des Dialogs	#PCDATA	?
<DialogLabel>	Bezeichnung des Dialogs, die ggf. in der Benutzeroberfläche verwendet wird	#PCDATA	?
<DialogType>	Art des Dialogs, z. B. modaler Dialog	Enumeration[225]	1
<Position>	Position des Dialogs in der Dialog-Abfolge	EMPTY	?
<Predecessors>	Liste von Dialogen, die den Dialog als Nachfolger haben	EMPTY	?
<Predecessor>	Einzelner vorhergehender Dialog	EMPTY	* (+)
<PRED_DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	? (1)
<PRED_DialogName>	Bezeichnung des Dialogs	#PCDATA	?
<PRED_TransitionType>	Art der Transition, d.h. sequenziell oder nebenläufig	Enumeration[226]	? (1)
<PRED_Trigger>	Auslöser für die Dialog-Transition	EMPTY	? (1)
<PRED_ConceptID>	Eindeutiger Bezeichner des auslösenden Konzepts	#PCDATA	? (1)
<PRED_ConceptName>	Bezeichnung des auslösenden Konzepts	#PCDATA	?
<Successors>	Liste von nachfolgenden Dialogen	EMPTY	?

[225] (Single, Multi, Modal, Complex)
[226] (Sequential | Concurrent)

Element	Beschreibung	Typ	Kardinalität
<Successor>	Einzelner nachfolgender Dialog	EMPTY	* (+)
<SUCC_DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	? (1)
<SUCC_DialogName>	Bezeichnung des Dialogs	#PCDATA	?
<SUCC_TransitionType>	Art der Transition, d.h. sequenziell oder nebenläufig	Enumeration[227]	? (1)
<SUCC_Trigger>	Auslöser für die Dialog-Transition	EMPTY	? (1)
<SUCC_ConceptID>	Eindeutiger Bezeichner des auslösenden Konzeots	#PCDATA	? (1)
<SUCC_ConceptName>	Bezeichnung des auslösenden Konzepts	#PCDATA	?
<DLG_Tasks>	Liste der Aufgaben aus dem Aufgaben-Modell, die in dem Dialog enthalten sind	EMPTY	?
<DLG_Task>	Angaben zu einer Aufgabe	EMPTY	* (+)
<DLG_TaskID>	Eindeutiger Bezeichner der Aufgabe	#PCDATA	? (1)
<DLG_TaskName>	Bezeichnung der Aufgabe	#PCDATA	?
<DLG_Processing>	Angabe, ob nur die Aufgabe selbst oder auch alle dieser hierarchisch untergeordneten Aufgaben Teil des Dialogs sind	Enumeration[228]	1
<SupplementaryConcepts>	Liste von zusätzlichen UI-Konzepten, für die keine Aufgabe im Task-Mo dell existiert	EMPTY	?
<SupplementaryConcept>	Spezifikation eines zusätzlichen UI-Konzepts	EMPTY	* (+)
<SupplementaryConceptID>	Eindeutiger Bezeichner des zusätzlichen UI-Konzepts im Konzept-Modell	#PCDATA	? (1)
<SupplementaryConceptName>	Bezeichnung des zusätzlichen UI-Konzepts	#PCDATA	?

[227] (Sequential | Concurrent)
[228] (Exclusive | Recursive)

E.4 Beschreibungselemente eines Benutzungskontext-Modells

Tabelle 68 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES KONTEXT-MODELLS <CONTEXTMODELREFERENCES>

Element	Beschreibung	Typ	Kardinalität
<ContextModelReferences>	Verweise auf die vier Teilmodelle	EMPTY	1
<UserModelReference	Verweis auf das zugehörige Benutzer-Modell	EMPTY	?
<USRREF_ModelID>	Eindeutiger Bezeichner des Benutzer-Modells	#PCDATA	? (1)
<USRREF_ModelVersion>	Versionsnummer des Benutzer-Modells	#PCDATA	? (1)
<USRREF_ModelRevision>	Revisionsnummer des Benutzer-Modells	#PCDATA	? (1)
<USRREF_ModelName>	Bezeichnung des Benutzer-Modells	#PCDATA	?
<DeviceModelReference>	Verweis auf das zugehörige Geräte-Modell	EMPTY	1
<DEVREF_ModelID>	Eindeutiger Bezeichner des Geräte-Modells	#PCDATA	1
<DEVREF_ModelVersion>	Versionsnummer des Geräte-Modells	#PCDATA	1
<DEVREF_ModelRevision>	Revisionsnummer des Geräte-Modells	#PCDATA	1
<DEVREF_ModelName>	Bezeichnung des Geräte-Modells	#PCDATA	?
<ToolkitModelReference	Verweis auf das zugehörige UI-Toolkit-Modell	EMPTY	1
<TLKREF_ModelID>	Eindeutiger Bezeichner des UI-Toolkit-Modells	#PCDATA	1
<TLKREF_ModelVersion>	Versionsnummer des UI-Toolkit-Modells	#PCDATA	1
<TLKREF_ModelRevision>	Revisionsnummer des UI-Toolkit-Modells	#PCDATA	1
<TLKREF_ModelName>	Bezeichnung des UI-Toolkit-Modells	#PCDATA	?
<EnvironmentModelReference	Verweis auf das zugehörige Umgebungs-Modell	EMPTY	?
<ENVREF_ModelID>	Eindeutiger Bezeichner des Umgebungs-Modells	#PCDATA	? (1)
<ENVREF_ModelVersion>	Versionsnummer des Umgebungs-Modells	#PCDATA	? (1)
<ENVREF_ModelRevision>	Revisionsnummer des Umgebungs-Modells	#PCDATA	? (1)
<ENVREF_ModelName>	Bezeichnung des Umgebungs-Modells	#PCDATA	?

E.4.1 Beschreibungselemente eines Benutzer-Modells

Tabelle 69 BESCHREIBUNGSELEMENTE EINES BENUTZER-MODELLS <USERMODEL>

Element	Beschreibung	Typ	Kardinalität
<UserCharacteristics>	Eigenschaften des Benutzers bzw. der Gruppe von Benutzern	EMPTY	1
<UserAbilities>	Fähigkeiten des Benutzers bzw. der Gruppe von Benutzern	EMPTY	1
<USRUA_Visual>	Sehvermögen	Enumeration[229]	1
<USRUA_Acoustic>	Hörvermögen	Enumeration[229]	1
<USRUA_Motor>	Motorische Fähigkeiten	Enumeration[229]	1
<USRUA_Mental>	Geistige Fähigkeiten	Enumeration[229]	1
<UserExperiences>	Erfahrungen, Fertigkeiten und Wissen des Benutzers bzw. der Gruppe von Benutzern	EMPTY	?
<USRUE_Domain>	Kenntnisse bzgl. ser Anwendungsdomäne	Enumeration[230]	? (1)
<USRUE_Handling>	Übung im Umgang mit elektronischen Geräten	Enumeration[230]	? (1)
<UserDistraction>	Grad der Abgelenktheit bei der Bedienung der Benutzeroberfläche	Enumeration[229]	?
<UserLegalCapacity>	Geschäftsfähigkeit des Benutzers bzw. der Gruppe von Benutzern	Enumeration[231]	?
<UserEmotionalState>	Informationen zum Gemütszustand des Benutzers	#PCDATA	?

[229] (High | Medium | Low)
[230] (Expert | Advanced | Novice)
[231] (Full | Limited | None)

E.4.2 Beschreibungselemente eines Geräte-Modells

Tabelle 70 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES GERÄTE-MODELLS <DEVICECHARACTERISTICS>

Element	Beschreibung	Typ	Kardinalität
<DeviceCharacteristics>	Eigenschaften des Endgeräts	EMPTY	1
<DeviceOperatingSystems>	Liste der für das Endgerät zur Verfügung stehenden Betriebssysteme	EMPTY	1
<DeviceOperatingSystem>	Bezeichnung eines Betriebssystems	#PCDATA	+
<DeviceInputFacilities>	Möglichkeiten zur Eingabe	EMPTY	1
<DEVIF_Motor>	Motorische Eingabemöglichkeiten	EMPTY	?[234]
<DEVIF_Keyboard>	Art der Tastatur	Enumeration[232]	? (1)
<DEVIF_TouchScreen>	Berührungsempfindlicher Bildschirm	Enumeration[233]	? (1)
<DEVIF_PointingDevice>	Zeigegerät, z. B. Maus oder Trackball	Enumeration[233]	? (1)
<DEVIF_Acoustic>	Akustische Eingabemöglichkeiten	EMPTY	?[234]
<DEVIF_VoiceControl>	Sprachsteuerung	Enumeration[233]	? (1)
<DEVIF_Visual>	Visuelle Eingabemöglichkeiten	EMPTY	?[234]
<DEVIF_GestureControl>	Gestensteuerung	Enumeration[233]	? (1)
<DEVIF_EyeTracking>	Blickverfolgung	Enumeration[233]	? (1)
DeviceOutputCapabilities>	Möglichkeiten zur Ausgabe	EMPTY	1
<DEVOF_Visual>	Visuelle Ausgabemöglichkeiten	EMPTY	?[235]
<DEVOF_Screen>	Bildschirm	EMPTY	?
<DEVOF_ScreenSize>	Größe des Bildschirms	#PCDATA	? (1)
<DEVOF_ScreenResolution>	Auflösung des Bildschirms	#PCDATA	? (1)
<DEVOF_Acoustic>	Akustische Ausgabemöglichkeiten	EMPTY	?[235]
<DEVOF_Speaker>	Lautsprecher	Enumeration[233]	? (1)
<DEVOF_Headphones>	Ohr- oder Kopfhörer	Enumeration[233]	? (1)

[232] (Hardware | Software | None)
[233] (Yes | No)
[234] Es muss wendigstens eine der genannten Eingabemöglichkeiten (motorisch, akustisch oder visuell) existieren
[235] Es muss wenigstens eine der genannten Ausgabemöglichkeiten (visuell, akustisch oder taktil) existieren

Element	Beschreibung	Typ	Kardinalität
<DEVOF_Tactile>	Taktile Ausgabemöglichkeiten	EMPTY	?[235]
<DEVOF_Vibration>	Vibrationssignale	Enumeration[233]	? (1)
<DEVOF_Braille>	Braille-Display oder Braille-Zeile	Enumeration[233]	? (1)

E.4.3 Beschreibungselemente eines UI-Toolkit-Modells

Tabelle 71 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES TOOLKIT-MODELLS <TOOLKITCHARACTERISTICS>

Element	Beschreibung	Typ	Kardinalität
<ToolkitCharacteristics>	Eigenschaften des UI-Toolkits	EMPTY	1
<ToolkitSupportedOSPlatforms>	Liste der Betriebssystem-Plattformen auf denen das UI-Toolkit verfügbar ist	EMPTY	1
<ToolkitSupportedOSPlatform>	Bezeichnung einer Betriebssystem-Plattform	#PCDATA	+
<ToolkitSupportedUIObjects>	Liste der UI-Objekte, die durch das UI-Toolkit zur Verfügung gestellt werden	EMPTY	1
<ToolkitSupportedUIObject>	Angaben zu einem UI-Objekt	EMPTY	+
<TLKSO_Name>	Bezeichnung des UI-Objekts	#PCDATA	1
<TLKSO_Description>	Textuelle Beschreibung des UI-Objekts	#PCDATA	?
<TLKSO_Diagram>	Beispielhafte Abbildung	ANY	?

E.4.4 Beschreibungselemente eines Umgebungs-Modells

Tabelle 72 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES TOOLKIT-MODELLS <ENVIRONMENTCHARACTERISTICS>

Element	Beschreibung	Typ	Kardinalität
<EnvironmentCharacteristics>	Eigenschaften der Umgebung, in der User bei der Benutzung des Systems befindet	EMPTY	1
<LuminousReflectance>	Beeinträchtigung durch Lichtreflexion	Enumeration[236]	1
<AmbientNoise>	Beeinträchtigung durch Umgebungsgeräusche	Enumeration[236]	1
<Pollutant>	Beeinträchtungen durch Staub und Schmutz	Enumeration[236]	1

[236] (High | Medium | Low)

E.5 Beschreibungselemente eines AUI-Modells

Tabelle 73 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES AUI-MODELLS <ABSTRACTUSERINTERFACE>

Element	Beschreibung	Typ	Kardinalität
<AbstractUserInterface>	Liste der in der abstrakten Benutzeroberfläche enthaltenen AUI-Elemente	EMPTY	1
<AUI_Element>	Angaben zu einem AUI-Element	EMPTY	+
<AUIE_ID>	Eindeutiger Bezeichner des AUI-Elements	#PCDATA	1
<AUIE_Name>	Bezeichnung des AUI-Elements	#PCDATA	1
<AUIE_Description>	Textuelle Beschreibung des AUI-Elements	#PCDATA	?
<AUIE_Label>	Bezeichnung des AUI-Elements, die in der Benutzeroberfläche verwendet wird	#PCDATA	?
<AUIE_Perceptible>	Festlegung, ob das AUI-Element durch den Benutzer wahrnehmbar (z. B. sichtbar) ist oder nicht	Enumeration[237]	1
<AUIE_Enabled>	Festlegung, ob das AUI-Element aktuell verwendet werden kann oder nicht	Enumeration[237]	1
<AUIE_Required>	Festlegung, ob das AUI-Element vom Benutzer verwendet werden muss oder nicht	Enumeration[237]	1
<AUIE_Type>	Art des AUI-Elements, z. B. DataInput, DataOutput oder DataEdit	Enumeration[238]	1
<AUIE_DataType>	Datentyp des AUI-Elements	Enumeration[239]	?
<AUIE_Value>	Wert	#PCDATA	? (1)
<AUIE_Origin>	Herkunft des AUI-Elements, d.h. die Konzepte, aus denen es transformiert wurde	EMPTY	1
<AUIE_OriginConcepts>	Liste der Herkunfts-Konzepte	EMPTY	1
<AUIE_OriginConcept>	Einzelnes Herkunfts-Konzept	EMPTY	+
<AUIE_ConceptID>	Eindeutiger Bezeichner dieses Herkunfts-Konzepts	#PCDATA	1
<AUIE_ConceptName>	Bezeichnung dieses Herkunfts-Konzepts	#PCDATA	?
<AUIE_Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor das AUI-Element verwendet werden kann	EMPTY	?

[237] (True | False)
[238] Siehe Tabelle 23
[239] (Integer | Float | String | Boolean | Picture)

Element	Beschreibung	Typ	Kardinalität
<AUIE_Precondition>	Einzelne Vorbedingung	EMPTY[240]	*
<AUIE_Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem das AUI-Element verwendet wurde	EMPTY	?
<AUIE_Postcondition>	Einzelne Nachbedingung	EMPTY[240]	*
<AUIE_DataLink>	Verweis auf ein Datenelement im Datenmodell der zugrunde liegenden Anwendung	EMPTY	?
<AUIE_DataLinkID>	Eindeutiger Bezeichner des Verweises	#PCDATA	? (1)
<AUIE_DataLinkName>	Bezeichnung des Verweises	#PCDATA	? (1)
<AUIE_DataLinkDescription>	Texuelle Beschreibung des Verweises	#PCDATA	?
<AUIE_DataLinkReference>	Referenz zum betreffenden Datenelement, z. B. der Name einer Klasse	#PCDATA	? (1)
<AUIE_ContainedAUIElements>	Liste von in Container-Elementen enthaltenen AUI-Elementen	EMPTY	?
<AUIE_ContainedAUIElement>	Angaben zu einem enthaltenen AUI-Element	<AUI_Element>	* (+)

[240] Weitergehende Beschreibungsstruktur siehe Tabelle 65

E.6 Beschreibungselemente eines CUI-Modells

Tabelle 74 SPEZIFISCHER ANTEIL DER BESCHREIBUNG EINES CUI-MODELLS <CONCRETEUSERINTERFACE>

Element	Beschreibung	Typ	Kardinalität
<ConcreteUserInterface>	Liste der in der konkreten Benutzeroberfläche enthaltenen CUI-Elemente	EMPTY	1
<CUI_Element>	Angaben zu einem CUI-Element	EMPTY	+
<CUIE_ID>	Eindeutiger Bezeichner des CUI-Elements	#PCDATA	1
<CUIE_Name>	Bezeichnung des CUI-Elements	#PCDATA	1
<CUIE_Description>	Textuelle Beschreibung des CUI-Elements	#PCDATA	?
<CUIE_Label>	Bezeichnung des CUI-Elements, die in der Benutzeroberfläche verwendet wird	#PCDATA	?
<CUIE_Perceptible>	Festlegung, ob das CUI-Element durch den Benutzer wahrnehmbar (z. B. sichtbar) ist oder nicht	Enumeration[241]	1
<CUIE_Enabled>	Festlegung, ob das CUI-Element aktuell verwendet werden kann oder nicht	Enumeration[241]	1
<CUIE_Required>	Festlegung, ob das CUI-Element vom Benutzer verwendet werden muss oder nicht	Enumeration[241]	1
<CUIE_Type>	Art des CUI-Elements, z. B. DataInput, DataOutput oder DataEdit	Enumeration[242]	1
<CUIE_DataType>	Datentyp des CUI-Elements	Enumeration[243]	?
<CUIE_Value>	Wert	#PCDATA	? (1)
<CUIE_Presentation>	Angaben zum Erscheinungsbild des CUI-Elements	EMPTY	?
<CUIE_Size>	Größe	EMPTY	?
<CUIE_Height>	Höhe	#PCDATA	? (1)
<CUIE_Width>	Breite	#PCDATA	? (1)
<CUIE_Position>	Position	EMPTY	?
<CUIE_HorizontalPosition>	Horizontale Position	#PCDATA	? (1)
<CUIE_VerticalPosition>	Vertikale Position	#PCDATA	? (1)
<CUIE_Orientation>	Ausrichtung	EMPTY	?

[241] (True | False)
[242] Siehe Tabelle 24
[243] (Integer | Float | String | Boolean | Picture)

Element	Beschreibung	Typ	Kardinalität
<CUIE_HorizontalOrientation>	Horizontale Ausrichtung	Enumeration[244]	? (1)
<CUIE_VerticalOrientation>	Vertikale Ausrichtung	Enumeration[245]	? (1)
<CUIE_Color>	Farbe	EMPTY	?
<CUIE_ForegroundColor>	Vordergrundfarbe	#PCDATA	? (1)
<CUIE_BackgroundColor>	Hintergrundfarbe	#PCDATA	? (1)
<CUIE_HighlightColor>	Hervorhebungsfarbe	#PCDATA	? (1)
<CUIE_Border>	Rahmen	EMPTY	?
<CUIE_BorderWidth>	Breite des Rahmens	#PCDATA	? (1)
<CUIE_BorderStyle>	Stilart des Rahmens	#PCDATA	? (1)
<CUIE_Origin>	Herkunft des CUI-Elements, d.h. die AUI-Elemente, aus denen es transformiert wurde	EMPTY	1
<CUIE_OriginAUIElements>	Liste der Herkunfts-AUI-Elemente	EMPTY	1
<CUIE_OriginAUIElement>	Einzelnes Herkunfts-AUI-Element	EMPTY	+
<CUIE_AUIElementID>	Eindeutiger Bezeichner dieses Herkunfts-AUI-Elements	#PCDATA	1
<CUIE_AUIElementName>	Bezeichnung dieses Herkunfts-AUI-Elements	#PCDATA	?
<CUIE_Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor das CUI-Element verwendet werden kann	EMPTY	?
<CUIE_Precondition>	Einzelne Vorbedingung	EMPTY[246]	*
<CUIE_Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem das CUI-Element verwendet wurde	EMPTY	?
<CUIE_Postcondition>	Einzelne Nachbedingung	EMPTY[246]	*
<CUIE_DataLink>	Verweis auf ein Datenelement im Datenmodell der zugrunde liegenden Anwendung	EMPTY	?
<CUIE_DataLinkID>	Eindeutiger Bezeichner des Verweises	#PCDATA	? (1)
<CUIE_DataLinkName>	Bezeichnung des Verweises	#PCDATA	? (1)
<CUIE_DataLinkDescription>	Texuelle Beschreibung des Verweises	#PCDATA	?

[244] (Left | Center | Right)
[245] (Top | Center | Bottom)
[246] Weitergehende Beschreibungsstruktur siehe Tabelle 65

Element	Beschreibung	Typ	Kardinalität
<CUIE_DataLinkReference>	Referenz zum betreffenden Datenelement, z. B. der Name einer Klasse	#PCDATA	? (1)
<CUIE_ContainedCUIElements>	Liste von in Container-Elementen enthaltenen CUI-Elementen	EMPTY	?
<CUIE_ContainedCUIElement>	Angaben zu einem enthaltenen CUI-Element	<CUI_Element>	* (+)

F. Beschreibung der Zuordnungstabellen für Modelltransformationen im Detail

In den folgenden Unterkapiteln werden die Beschreibungselemente der vier Top-Level-Elemente <Head>, <Body>, <Relationship> und <Deployment> im Detail beschrieben. Es ist jeweils der Name des Beschreibungselements, eine kurze Beschreibung, der Datentyp und die Kardinalittät angegeben.

Die Festlegung der Datentypen erfolgt gemäß folgender Konventionen:

- EMPTY Das Element ist immer leer und dienst nur zu Strukturierungszwecken
- #PCDATA Das Element enthält ausschließlich Text
- ANY Das Element kann beliebigen Inhalt haben
- Enumeration Das Element kann einen der in der betreffenden Fußnote spezifizierten Werte annehmen

Die Spezifikation der Kardinalität erfolgt gemäß folgender Konventionen:

- 1 Das Element ist verpflichtend und existiert genau einmal
- ? Das Element ist optional und existiert nicht oder genau einmal
- * Das Element tritt keinmal, einmal oder beliebig häufig auf
- + Das Element tritt mindestens einmal oder beliebig häufig auf

Teilweise werden zwei Angaben zur Kardinalität gemacht, wobei die zweite von runden Klammern umschlossen ist, z. B. „? (1)“. Hierbei spezifiziert die erste Angabe die generelle Kardinalität und der in Klammern angegebene Wert die Kardinalität in Abhängigkeit der Existenz des direkt übergeordneten Elements. Das genannte Beispiel ist demnach wie folgt zu lesen:

- ? Das zugehörige Element tritt generell keinmal oder genau einmal auf
- (1) Falls das übergeordnete Element existiert, also die Kardinalität „1“ besitzt, dann muss auch dieses Element exisitieren

F.1 Beschreibungselemente von Zuordnungstabellen für Modelltransformationen

Tabelle 75 BESCHREIBUNGSELEMENTE VON PAMGIS-ZUUORDNUNGSTABELLEN FÜR MODELLTRANSFORMATIONEN <PAMGIS_MAPPINGTABLE>

Element	Beschreibung	Typ	Kardinalität
<UniqueMappingTableID>	Eindeutiger Bezeichner der Zuordungstabelle	EMPTY	1
<MappingTableID>	Bezeichner der Zuordnungstabelle	#PCDATA	1
<MappingTableVersion>	Versionsnummer der Zuordnungstabelle, i.d.R. zweistellig, wird nach signifikanten Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<MappingTableRevision>	Revisionsnummer der Zuordnungstabelle, i.d.R. zweistellig, wird nach kleineren Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<MappingTableType>	Typ der Zuordungstabelle, z. B. AUI2CUI	Enumeration[247]	1
<MappingTableName>	Bezeichnung der Zuordungstabelle	#PCDATA	1
<MappingTableDescription>	Textuelle Beschreibung der Zuordungstabelle	#PCDATA	?
<MappingTableAnnotation>	Anmerkungen und Hinweise in Textform	#PCDATA	?
<MappingTableUserComments>	Liste von Benutzerkommentaren	EMPTY	?
<MappingTableUserComment>	Kommentar eines Benutzers	EMPTY	* (+)
<MTBUC_Author>	Name des Erstellers	#PCDATA	? (1)
<MTBUC_Date>	Datum	#PCDATA	? (1)
<MTBUC_Comment>	Kommentar des Benutzers	#PCDATA	? (1)
<ApplicableContextsOfUse>	Liste von referenzierten Kontext-Modellen, für die die Zuordnungstabelle verwendet werden kann	EMPTY	?
<ApplicableContextOfUse>	Einzelnes referenziertes Kontext-Modell	EMPTY	? (+)
<COURFF_ModelID>	Eindeutiger Bezeichner des Kontext-Modells	#PCDATA	? (1)
<COUREF_ModelVersion>	Versionsnummer des Kontext-Modells	#PCDATA	? (1)
<COUREF_ModelRevision>	Revisionsnummer des Kontext-Modells	#PCDATA	? (1)
<COUREF_ModelName>	Bezeichnung des Kontext-Modells	#PCDATA	?

[247] (DOM2AUI |AUI2CUI | CUI2FUI)

Element	Beschreibung	Typ	Kardinalität
<MappingRules>	Liste der Zuordnungsregeln	EMPTY	1
<MappingRule>	Einzelne Zuordnungregel	EMPTY	+
<RuleSequentialNumber>	Laufende Nummer der Zuordnungsregel	#PCDATA	1
<MappingSource>	Elementtyp des Quellmodells, der transformiert werden soll	#PCDATA	1
<MappingCondition>	Bedingung, unter der die Transformation ausgeführt werden soll	#PCDATA	?
<MappingTarget>	Elementtyp des Zielmodells nach der Transformation	#PCDATA	1
<RuleAnnotation>	Anmerkungen und Hinweise zur Zuordnungsregel in Textform	#PCDATA	?

G. Beschreibung von PPSL im Detail

In den folgenden Unterkapiteln werden die Beschreibungselemente der vier Top-Level-Elemente <Head>, <Body>, <Relationship> und <Deployment> im Detail beschrieben. Es ist jeweils der Name des Beschreibungselements, eine kurze Beschreibung, der Datentyp und die Kardinalittät angegeben.

Die Festlegung der Datentypen erfolgt gemäß folgender Konventionen:

- EMPTY Das Element ist immer leer und dienst nur zu Strukturierungszwecken
- #PCDATA Das Element enthält ausschließlich Text
- ANY Das Element kann beliebigen Inhalt haben
- Enumeration Das Element kann einen der in der betreffenden Fußnote spezifizierten Werte annehmen

Die Spezifikation der Kardinalität erfolgt gemäß folgender Konventionen:

- 1 Das Element ist verpflichtend und existiert genau einmal
- ? Das Element ist optional und existiert nicht oder genau einmal
- * Das Element tritt keinmal, einmal oder beliebig häufig auf
- + Das Element tritt mindestens einmal oder beliebig häufig auf

Teilweise werden zwei Angaben zur Kardinalität gemacht, wobei die zweite von runden Klammern umschlossen ist, z. B. „? (1)“. Hierbei spezifiziert die erste Angabe die generelle Kardinalität und der in Klammern angegebene Wert die Kardinalität in Abhängigkeit der Existenz des direkt übergeordneten Elements. Das genannte Beispiel ist demnach wie folgt zu lesen:

- ? Das zugehörige Element tritt generell keinmal oder genau einmal auf
- (1) Falls das übergeordnete Element existiert, also die Kardinalität „1“ besitzt, dann muss auch dieses Element exisitieren

G.1 Liste der Beschreibungselemente von PPSL

Tabelle 76 TOP-LEVEL-ELEMENTE VON PPSL

Element	Beschreibung	Typ	Kardinalität
<Head>	Pattern-bezogene Metadaten, u.a. Informationen zur Identifikation des Musters, Namen, Bearbeitungsstatus, Zusammenfassung, Urheber, Änderungshistorie, Quellenangaben und rechtliche Aspekte	EMPTY	1
<Body>	Eigentliche Beschreibung des Musters, aufgeteilt in theoretische und praktische Aspekte	EMPTY	1
<Relationship>	Beschreibung von Beziehungen zu anderen Mustern	EMPTY	?
<Deployment>	Informationen zur Implementierung der Musters, die u.a. automatisiert verarbeitet werden können, z. B. Code-Fragmente oder Modell-Fragmente, die von PaMGIS verarbeitet werden können	EMPTY	?

G.1.1 Beschreibungselemente im Top-Level-Element <Head>

Tabelle 77 IM TOP-LEVEL-ELEMENT <HEAD> ZUSAMMENGEFASSTE BESCHREIBUNGSELEMENTE VON PPSL

Element	Beschreibung	Typ	Kardinalität
<Texture>	Verwendete Version von PPSL	#PCDATA	1
<UniquePatternID>	Eindeutiger Bezeichner des Musters (engl. unique pattern identifier)	EMPTY	1
<UPID_PatternCompilationID>	Eindeutiger Bezeichner der Pattern-Sammlung, zu der das Muster gehört (engl. pattern compilation identifier)	#PCDATA	1
<UPID_PatternID>	Eindeutiger Bezeichner des Musters (engl. pattern identifier)	#PCDATA	1
<UPID_VersionNumber>	Versionsnummer des Musters, i.d.R. zweistellig, wird nach signifikanten Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<UPID_RevisionNumber>	Revisionsnummer des Musters, i.d.R. zweistellig, wird nach kleineren Änderungen oder Ergänzungen um den Wert „1“ erhöht	#PCDATA	1
<Classification>	Information zur Klassifikation des Musters	EMPTY	1
<CLSS_PatternCompilationName>	Name der Pattern-Sammlung, zu der das Muster gehört	#PCDATA	1
<CLSS_Domains>	Liste von Domänen, in denen das Muster angewendet werden kann	EMPTY	?
<CLSS_Domain>	Name einer Domäne, in der das Muster angewendet werden kann	#PCDATA	* (+)
<CLSS_PatternType>	Art des Musters, z. B. „HCI Design Pattern“	#PCDATA	1

Element	Beschreibung	Typ	Kardinalität
<CLSS_AbstractionLevel>	Abstraktionsgrad des Musters	#PCDATA	?
<CLSS_Category>	Kategorie, der das Muster in der Pattern-Sammlung zugeordnet ist	#PCDATA	?
<CLSS_Subcategory>	Unterkategorie, der das Muster ggf. zugeordnet ist	#PCDATA	?
<CLSS_SubSubCategory>	Unterkategorie der Unterkategorie, der das Muster ggf. zugeordnet ist	#PCDATA	?
<CLSS_Keywords>	Liste von Schlüsselwörtern, die dem Muster zugeordnet sind	EMPTY	?
<CLSS_Keyword>	Einzelnes Schlüsselwort, das dem Muster zugeordnet ist	#PCDATA	* (+)
<Names>	Bezeichnungen für das Muster	EMPTY	1
<Name>	Aussagekräftige Bezeichnung des Musters	#PCDATA	1
<Aliases>	Liste alternativer Bezeichnungen, unter der das Muster auch bekannt ist	EMPTY	?
<Alias>	Einzelne alternative Bezeichnung, unter der das Muster bekannt ist	#PCDATA	* (+)
<Status>	Bearbeitungsstand der Pattern-Spezifikation, z. B. “Vorschlag”, “in Bearbeitung”, “intern freigegeben”, “allgemein freigegeben“ oder “zurückgezogen”. Intern freigegebene Muster sind für registrierte Benutzern und Power-User sichtbar, generell freigegebene auch für anonyme Benutzer.	Enumeration[248]	1
<Synopsis>	Prägnante Zusammenfassung von Bedeutung und Inhalt des Musters	#PCDATA	?
<Contribution>	Informationen zu Personen, die zur Beschreibung des Musters beigetragen haben	EMPTY	1
<Authors>	Liste der an der Pattern-Spezifikation beteiligten Autoren	EMPTY	1
<Author>	Angaben zu einem Autor	EMPTY	+
<ATHR_AuthorID>	Eindeutiger Bezeichner des Autors	#PCDATA	1
<ATHR_FirstName>	Vorname des Autors	#PCDATA	1
<ATHR_LastName>	Nachname des Autors	#PCDATA	1
<ATHR_Organization>	Name der Organisation, der der Autor angehört, z. B. eine Universität	#PCDATA	?
<ATHR_PersonalData>	Informationen über die Person, z. B. Lebenslauf	ANY	?
<ATHR_ContactInformation>	Kontaktdaten des Autors, z. B. Anschrift und/oder Email-Adresse	#PCDATA	?
<ATHR_Link>	Link auf eine Webseite des Autors, z. B. eine persönliche Homepage	EMPTY	?
<ATHR_LinkDisplayName>	Benutzerfreundliche Bezeichnung des Links	#PCDATA	?
<ATHR_URL>	Web-Adresse (engl. uniform resource locator)	#PCDATA	? (1)

[248] (Proposed | In progress | Internally released | Generally released | Withdrawn)

Element	Beschreibung	Typ	Kardinalität
<Credits>	Angaben zu Verdiensten von Personen, die zur Spezifikation des Musters beigetragen haben, aber nicht als Autor genannt sind	#PCDATA	?
<History>	Historische Entwicklung der Pattern-Spezifikation	EMPTY	1
<Ancestry>	Information über die Herkunft des Musters (falls zustreffend)	EMPTY	?
<ANCS_PatternCompilationName>	Name der Pattern-Sammlung, aus der das Muster übernommen wurde	#PCDATA	? (1)
<ANCS_PatternName>	Name des Musters in dieser Pattern-Sammlung	#PCDATA	? (1)
<ANCS_Texture>	Name und Version der verwendeten Pattern-Beschreibungssprache	#PCDATA	?
<ANCS_PatternID>	Eindeutiger Bezeichner des Musters in dieser Pattern-Sammlung	#PCDATA	?
<ANCS_Category>	Kategorie, der das Muster in dieser Pattern-Sammlung zugeordnet ist	#PCDATA	?
<ANCS_VersionNumber>	Versionsnummer des Musters	#PCDATA	?
<ANCS_RevisionNumber>	Revisionsnummer des Musters	#PCDATA	?
<ANCS_Link>	Link auf eine Webseite mit Informationen zu dem Muster	EMPTY	?
<ANCS_LinkDisplayName>	Benutzerfreundliche Bezeichnung des Links	#PCDATA	?
<ANCS_URL>	Web-Adresse (engl. uniform resource locator)	#PCDATA	? (1)
<CreationDate>	Datum der Erstellung des Musters mithilfe von PPSL	#PCDATA	1
<LastModified>	Datum der letzten Änderung des Musters	#PCDATA	1
<Changes>	Liste von vorgenommenen Änderungen	EMPTY	?
<Change>	Angaben zu einer Änderung	EMPTY	* (+)
<CHNG_ChangeID>	Eindeutiger Bezeichner der Änderung	#PCDATA	? (1)
<CHNG_Date>	Datum der Änderung	#PCDATA	? (1)
<CHNG_VersionNumber>	Versionsnummer des Musters vor der Änderung	#PCDATA	? (1)
<CHNG_RevisionNumber>	Revisionsnummer des Musters vor der Änderung	#PCDATA	? (1)
<CHNG_Originator>	Name(n) oder ID(s) des oder der Autoren	#PCDATA	? (1)
<CHNG_Description>	Beschreibung der Änderung	#PCDATA	? (1)
<CHNG_Reason>	Begründung für und Hintergrundinformation zur Änderung	#PCDATA	? (1)
<CHNG_Link>	Link auf eine Webseite mit weitergehenden Informationen	EMPTY	?
<CHNG_LinkDisplayName>	Benutzerfreundliche Bezeichnung des Links	#PCDATA	?
<CHNG_URL>	Web-Adresse (engl. uniform resource locator)	#PCDATA	? (1)
<References>	Liste mit Referenzen (Quellen, Veröffentlichungen oder andere Literatur)	EMPTY	?
<Reference>	Angaben zu einer Referenz	EMPTY	* (+)
<RFNC_Type>	Art der Literatur, z. B. Buch oder Webseite	#PCDATA	? (1)
<RFNC_Title>	Titel oder Bezeichnung	#PCDATA	? (1)
<RFNC_Author>	Name(n) oder ID(s) des oder der Verfasser	#PCDATA	? (1)

Element	Beschreibung	Typ	Kardinalität
<RFNC_Description>	Knappe Angabe zu Zweck und Inhalt, z. B. „weitere Details“	#PCDATA	?
<RFNC_Reference>	Eindeutiger Bezeichner, z. B. ISBN oder URL	#PCDATA	? (1)
<LegalFoundation>	Rechtliche Grundlagen zur Verwendung des Musters	EMPTY	?
<Copyright>	Informationen zum urherrechtlichen Schutz	#PCDATA	?
<License>	Lizenz-Informationen	EMPTY	?
<LCNS_Description>	Beschreibung der Lizensierung	#PCDATA	? (1)
<LCNS_Type>	Lizenz-Typ, z. B. GNU	#PCDATA	?
<LCNS_Agreement>	Lizenzvereinbarung oder Verweis auf diese	#PCDATA	? (1)
<FurtherInformation>	Weitergehende Informationen, die sich keinem anderen Beschreibungselement zuordnen lassen	ANY	?

G.1.2 Beschreibungselemente im Top-Level-Element <Body>

Tabelle 78 IM TOP-LEVEL-ELEMENT <BODY> ZUSAMMENGEFASSTE BESCHREIBUNGSELEMENTE VON PPSL

Element	Beschreibung	Typ	Kardinalität
<Theory>	Theoretische Grundlagen zum vorliegenden Muster	EMPTY	1
<Problem>	Beschreibung des zu lösenden Problems	EMPTY	1
<PRBL_Digest>	Kurzfassung	#PCDATA	1
<PRBL_Elaboration>	Detaillierte Ausarbeitung	ANY	?
<Context>	Situationen und Umstände, in denen das Muster angewendet werden kann	EMPTY	1
<CNTX_Digest>	Kurzfassung	#PCDATA	1
<CNTX_Elaboration>	Detaillierte Ausarbeitung	ANY	?
<Solution>	Beschreibung, wie das beschriebene Problem im genannten Kontext gelöst werden kann	EMPTY	1
<SLTN_Digest>	Kurzfassung	#PCDATA	1
<SLTN_Elaboration>	Detaillierte Ausarbeitung	ANY	?
<Forces>	Liste der auf das Problem und die Lösung einwirkenden Einflüsse, die durch die Anwendung des Musters behoben werden	EMPTY	?
<Force>	Ein relevanter Einfluss	EMPTY	* (+)

Element	Beschreibung	Typ	Kardinalität
<FRCE_Digest>	Kurzfassung	#PCDATA	? (1)
<FRCE_Elaboration>	Detaillierte Ausarbeitung	ANY	?
<Rationale>	Diskussion und Begründung dafür, dass die Lösung das beschriebene Problem im genannten Kontext tatsächlich löst	#PCDATA	?
<Confidence>	Einschätzung der Wahrscheinlichkeit, dass das Muster eine invariante Lösung des beschriebenen Problems im genannten Kontext bietet	#PCDATA	?
<ResultingContext>	Kontext, der sich nach der Anwendung des Musters ergibt	#PCDATA	?
<Diagrams>	Liste schematischer Veranschaulichungen des Musters	EMPTY	?
<Diagram>	Eine schematisch Veranschaulichung	ANY	* (+)
<Practice>	Praktische Gesichtspunkte zur Verwendung des Musters	EMPTY	1
<Examples>	Liste von Anwendungsbeispielen (in PaMGIS)	EMPTY	?
<Example>	Angaben zu einem Anwendungsbeispiel	EMPTY	* (+)
<XMPL_Type>	Art des Anwendungsbeispiels (Beispiel oder Gegenbeispiel)	Enumeration[249]	? (1)
<XMPL_ExampleID>	Eindeutiger Bezeichner	#PCDATA	? (1)
<XMPL_Label>	Bezeichnung des Beispiels	#PCDATA	? (1)
<XMPL_Description>	Beschreibung des Beispiels	#PCDATA	? (1)
<XMPL_Diagram>	Schematische Veranschaulichung des Beispiels	ANY	?
<XMPL_Images>	Liste von Abbildungen, z. B. Screenshots	EMPTY	?
<XMPL_Image>	Eine relevante Abbildung	ANY	* (+)
<XMPL_Realizations>	Liste von Verweisen auf Implementierungen	EMPTY	?
<XMPL_Realization>	Verweis auf eine Implementierung, z. B. „ImplementationID“ aus dem PPSL-Top-Level-Element „Deployment“	#PCDATA	* (+)
<XMPL_Modelings>	Liste von Modellierungen	EMPTY	?
<XMPL_Modeling>	Liste von Verweisen auf zusammengehörende Modell-Fragmente aus dem PPSL-Top-Level-Element „Deployment“	EMPTY	* (+)
<XMPL_ModelFragmentID>	Eindeutiger Bezeichner eines Modell-Fragments	#PCDATA	* (+)
<KnownUses>	Liste von bekannten Verwendungen des Musters (außerhalb von	EMPTY	? (1)

[249] (Example | Counterexample)

Element	Beschreibung	Typ	Kardinalität
	PaMGIS)		
<KnownUse>	Angaben zu einer bekannten Verwendung	EMPTY	* (+)
<KNWN_CreationDate>	Datum der Erstellung	#PCDATA	?
<KNWN_KnownUseID>	Eindeutiger Bezeichner	#PCDATA	? (1)
<KNWN_Label>	Bezeichnung	#PCDATA	? (1)
<KNWN_Description>	Beschreibung	#PCDATA	? (1)
<KNWN_Originator>	Name des Urhebers	#PCDATA	?
<KNWN_Link>	Link auf eine Webseite auf der das Muster verwendet wird	EMPTY	?
<KNWN_LinkDisplayName>	Benutzerfreundliche Bezeichnung des Links	#PCDATA	?
<KNWN_URL>	Web-Adresse (engl. uniform resource locator)	#PCDATA	? (1)
<KNWN_Images>	Liste von Abbildungen, z. B. Screenshots	EMPTY	?
<KNWN_Image>	Eine relevante Abbildung	ANY	* (+)
<UserComments>	Liste von Kommentaren von Benutzern	EMPTY	?
<UserComment>	Kommentar eines Benutzers zum Pattern	EMPTY	* (+)
<UCMT_AuthorName>	Name des Erstellers	#PCDATA	? (1)
<UCMT_Date>	Erstellungsdatum	#PCDATA	? (1)
<UCMT_Comment>	Eigentlicher Kommentar des Benutzers	#PCDATA	? (1)
<UsabilityFeedback>	Ergebnisse von Usability-Aktivitäten, z. B. User-Testing	ANY	?

G.1.3 Beschreibungselemente im Top-Level-Element <Relationship>

Tabelle 79 IM TOP-LEVEL-ELEMENT <RELATIONSHIP> ZUSAMMENGEFASSTE BESCHREIBUNGSELEMENTE VON PPSL

Element	Beschreibung	Typ	Kardinalität
<RLTN_Information>	Beschreibung der Beziehungen zu anderen Mustern in Textform	ANY	?
<RLTN_Diagram>	Schematische Veranschaulichung der Beziehungen	ANY	?
<Relations>	Liste von Beziehungen zu anderen Mustern	EMPTY	?
<Relation>	Angaben zu einer Beziehung	EMPTY	* (+)
<RLTN_RelationID>	Eindeutiger Bezeichner der Relation	#PCDATA	? (1)
<RLTN_Nature>	Charakter der Beziehung (innerhalb von PaMGIS oder extern)	Enumeration[250]	? (1)
<RLTN_Type>	Art der Beziehung, z. B. „Assoziation“ oder „Komposition“	Enumeration[251]	? (1)
<RLTN_Sense>	Bedeutung der Beziehung, z. B. „besteht aus“ oder „ist Teil von“	Enumeration[252]	? (1)
<RLTN_Reference>	Identität des in Beziehung stehenden Musters	EMPTY	?
<RLTN_PatternCompilationID>	Eindeutiger Bezeichner der betreffenden Pattern-Sammlung	#PCDATA	?[253]
<RLTN_PatternCompilationName>	Name der betreffenden Pattern-Sammlung	#PCDATA	? (1)
<RLTN_PatternID>	Eindeutiger Bezeichner des betreffenden Musters	#PCDATA	?[253]
<RLTN_VersionNumber>	Versionsnummer des betreffenden Musters	#PCDATA	?
<RLTN_RevisionNumber>	Revisionsnummer des betreffenden Musters	#PCDATA	?
<RLTN_PatternName>	Name des betreffenden Musters	#PCDATA	? (1)
<RLTN_Description>	Beschreibung, Anmerkungen und Hinweise in Textform	#PCDATA	?

[250] (Internal | External)
[251] (Assoziation | Komposition)
[252] (nutzt | wird genutzt durch |besteht aus | ist Teil von)
[253] Falls es sich um eine Beziehung zu einem Muster handelt, das innerhalb von PaMGIS und mithilfe von PPSL spezifiziert ist (<RLTN_Nature> = „Internal“), dann müssen sowohl <RLTN_PatternCompilationID> als auch <RLTN_PatternID> verpflichtend angegeben werden.

G.1.4 Beschreibungselemente im Top-Level-Element <Deployment>

Tabelle 80 IM TOP-LEVEL-ELEMENT <DEPLOYMENT> ZUSAMMENGEFASSTE BESCHREIBUNGSELEMENTE VON PPSL

Element	Beschreibung	Typ	Kardinalität
<Implementations>	Liste von möglichen Umsetzungen der beschriebenen Lösung	EMPTY	?
<Implementation>	Angaben zu einer möglichen Umsetzung	EMPTY	* (+)
<IMPL_ImplementationID>	Eindeutiger Bezeichner der Umsetzung	#PCDATA	? (1)
<IMPL_Label>	Bezeichnung der Umsetzung	#PCDATA	? (1)
<IMPL_Description>	Beschreibung der Umsetzung	ANY	? (1)
<IMPL_Code>	Code-Fragment(e)	ANY	?
<EmbeddingLinks>	Liste von Rückverweisen auf OOA- und/oder OOD-Modellen [204]	EMPTY	?
<EmbeddingLink>	Angaben zu einem Rückverweis	EMPTY	* (+)
<EMBL_EmbeddedLinkID>	Eindeutiger Bezeichner des Rückverweises	#PCDATA	? (1)
<EMBL_Label>	Bezeichnung des Rückverweises	#PCDATA	? (1)
<EMBL_ReferenceClassID>	Klasse des OO-Modells, das das Muster referenziert	#PCDATA	? (1)
<EMBL_UMLRelationshipType>	Typ der Beziehung zwischen der betreffenden Klasse und dem Muster	#PCDATA	? (1)
<PaMGIS>	Für die Verwendung in PaMGIS vorgesehene Informationen	EMPTY	?
<ModelFragments>	Liste von Modell-Fragementen	EMPTY	? (1)
<ModelFragment>	Angaben zu einem Modell-Fragment	EMPTY	* (+)
<MDFR_Type>	Typ des Modell-Fragments, z. B. Task, Concept, Dialog	Enumeration[254]	? (1)
<MDFR_FragmentID>	Eindeutiger Bezeichner des Modell-Fragments	#PCDATA	? (1)
<MDFR_Label>	Bezeichnung des Modell-Fragements	#PCDATA	? (1)
<MDFR_Purpose>	Beschreibung des Zwecks, zu dem das Modell-Fragement dient	#PCDATA	?
<MDFR_Annotation>	Anmerkungen und Hinweise in Textform	#PCDATA	?
<MDFR_Diagram>	Schematische Veranschaulichung des Modell-Fragments	ANY	?
<MDFR_Fragment>	Inhalt des eigentlichen Modell-Fragments in Abhängigkeit des angegebenen Typs des Modell-Fragments (siehe Kapitel 4.7.2.2)	#PCDATA	? (1)

[254] (Task | Concept | Dialog)

G.2 Liste der Beschreibungselemente der Modell-Fragmente in PPSL

G.2.1 Beschreibungselemente eines Aufgaben-Modell-Fragments

Tabelle 81 BESCHREIBUNGSSELEMENTE VON PPSL FÜR TASK-MODELL-FRAGMENTE

Element	Beschreibung	Typ	Kardinalität
<TMF_IncludesDummy>	Kennzeichnung, ob das Task-Modell-Fragment einen oder mehrere Dummy-Tasks zur interaktiven Gestaltung variabler Modellanteile einhält	Enumeration[255]	?
<TMF_ProposedTempOp>	Vorschlag für die zeitliche Beziehung zwischen dem Wurzel-Element des Aufgaben-Modell-Fragments und der potenziell vorhergehenden Aufgabe im Aufgaben-Modell	Enumeration[256]	?
<TMF_Content>	Inhalt des Aufgaben-Modell-Fragments	EMPTY	?
<Subtask>	Wurzel-Element des Aufgaben-Modell-Fragments	EMPTY	? (1)
<TaskID>	Eindeutiger Bezeichner der Aufgabe	#PCDATA	? (1)
<TaskIDOriginal>	Ursprünglicher Bezeichner der Aufgabe, falls dieser z. B. bei der Anwendung eines Patterns geändert wurde	#PCDATA	?
<TaskName>	Bezeichnung der Aufgabe	#PCDATA	? (1)
<TaskDescription>	Textuelle Beschreibung der Aufgabe	#PCDATA	?
<TaskType>	Art der Aufgabe (abstrakte, Benutzer-, Interaktions-, Anwendungs- oder Dummy-Aufgabe)	Enumeration[257]	? (1)
<TaskOrigin>	Herkunft der Aufgabe, d.h. Pattern, das das Aufgaben-Modell-Fragment enthält	EMPTY	?
<TaskOriginPatternID>	Eindeutiger Bezeichner des Musters	#PCDATA	? (1)
<TaskOriginPatternVersion>	Versionsnummer des Musters	#PCDATA	? (1)
<TaskOriginPatternRevision>	Revisionsnummer des Musters	#PCDATA	? (1)
<TaskOriginPatternName>	Bezeichnung des Musters	#PCDATA	? (1)
<ContextsOfUse>	Liste von Benutzungskontexten, in denen die Aufgabe vorkommt	EMPTY	?

[255] (True | False)
[256] (SequentialEnabling | Disabling | Choice | Interleaving | Synchronization | SuspendResume | SequentialEnablingInfo)
[257] (Abstraction | User | Interaction | Application | Dummy)

Element	Beschreibung	Typ	Kardinalität
<ContextOfUse>	Bestimmung eines Benutzungskontext-Modells	EMPTY	* (+)
<ContextID>	Eindeutiger Bezeichner des Kontext-Modells	#PCDATA	?
<ContextVersion>	Versionsnummer des Kontext-Modells	#PCDATA	?
<ContextRevision>	Revisionsnummer des Kotext-Modells	#PCDATA	?
<ContextName>	Bezeichnung des Kontexts	#PCDATA	?
<ContextDescription>	Textuelle Beschreibung des Kontexts	#PCDATA	?
<Optional>	Kennzeichnung, ob die Task-Ausführung optional ist (wahr oder falsch)	Enumeration[258]	? (1)
<Iterative>	Kennzeichnung, ob die Aufgabe mehrfach ausgeführt werden kann (wahr oder falsch)	Enumeration[258]	? (1)
<Conditional>	Kennzeichnung, ob eine Bedingung hinsichtlich der Ausführung der Aufgabe vorliegt	Enumeration[258]	? (1)
<Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor die Aufgabe ausgeführt werden kann	EMPTY	?
<Precondition>	Einzelne Vorbedingung	EMPTY[259]	* (+)
<Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem die Aufgabe ausgeführt wurde	EMPTY	?
<Postcondition>	Einzelne Nachbedingung	EMPTY[259]	*
<Position>	Position der Aufgabe im Aufgaben-Modell	EMPTY	? (1)
<Parent>	Spezifikation der direkt übergeordneten Aufgabe	#PCDATA	? (1)
<ParentID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ beim Wurzel-Element	#PCDATA	?
<ParentName>	Bezeichnung dieser Aufgabe	EMPTY	? (1)
<SiblingLeft>	Spezifikation der vorhergehenden Aufgabe	#PCDATA	? (1)
<SiblingLeftID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ falls nicht vorhanden	#PCDATA	?
<SiblingLeftName>	Bezeichnung dieser Aufgabe	EMPTY	? (1)
<SiblingRight>	Spezifikation der nachfolgenden Aufgabe	#PCDATA	? (1)
<SiblingRightID>	Eindeutiger Bezeichner dieser Aufgabe oder „“ falls nicht vorhanden	#PCDATA	?
<SiblingRightName>			

[258] (True | False)
[259] Weitergehende Beschreibungsstruktur siehe Tabelle 65

Element	Beschreibung	Typ	Kardinalität
<TemporalOperator>	Bezeichnung dieser Aufgabe	Enumeration[260]	?
<UIConcepts>	Zeitliche Beziehung zur nachfolgenden Aufgabe	EMPTY	?
<UIConcept>	Liste der UI-Konzepte, die für die Ausführung der Aufgabe benötigt werden	EMPTY	* (+)
<UIConceptID>	Spezifikation eines UI-Konzepts	#PCDATA	? (1)
<UIConceptName>	Eindeutiger Bezeichner des UI-Konzepts	#PCDATA	?
<SubTasks>	Bezeichnung des UI-Konzepts	EMPTY	?
<SubTask>	Liste von Unteraufgaben	<Task>	* (+)
	Angaben zu einer Unteraufgabe		

G.2.2 Beschreibungselemente eines Konzept-Modell-Fragments

Tabelle 82 BESCHREIBUNGSELEMENTE VON PPSL FÜR KONZEPT-MODELL-FRAGMENTE

Element	Beschreibung	Typ	Kardinalität
<CMF_IncludesDummy>	Kennzeichnung, ob das Konzept-Modell-Fragment einen oder mehrere Dummy-Konzepte zur interaktiven Gestaltung variabler Modellanteile einhält	Enumeration[261]	?
<CMF_Content>	Inhalt des Konzept-Modell-Fragments	EMPTY	?
<Concept>	Angaben zu einem Konzept	EMPTY	* (+)
<CCPT_ConceptID>	Eindeutiger Bezeichner des Konzepts	#PCDATA	? (1)
<CCPT_ConceptIDOriginal>	Ursprünglicher Bezeichner der Aufgabe, falls dieser z. B. bei der Anwendung eines Patterns geändert wurde	#PCDATA	?
<CCPT_ConceptName>	Bezeichnung des Konzepts	#PCDATA	? (1)
<CCPT_Description>	Texutelle Beschreibung des Konzepts	#PCDATA	?
<CCPT_Label>	Bezeichnung des Konzepts, die in der Benutzeroberfläche verwendet wird	#PCDATA	?
<CCPT_Perceptible>	Festlegung, ob das Konzept durch den Benutzer wahrnehmbar (z. B. sichtbar)	Enumeration[262]	? (1)

[260] (SequentialEnabling | Disabling | Choice | Interleaving | Synchronization | SuspendResume | SequentialEnablingInfo)
[261] (True | False)

Element	Beschreibung	Typ	Kardinalität
	ist oder nicht		
<CCPT_Enabled>	Festlegung, ob das Konzept aktuell verwendet werden kann oder nicht	Enumeration[262]	? (1)
<CCPT_Required>	Festlegung, ob das Konzept vom Benutzer verwendet werden muss oder nicht	Enumeration[262]	? (1)
<CCPT_ConceptType>	Art des Konzepts, z. B. DataInput, DataOutput oder DataEdit	Enumeration[263]	? (1)
<CCPT_DataType>	Datentyp des Konzepts	#PCDATA	?
<CCPT_Value>	Wert	#PCDATA	? (1)
<CCPT_ConceptOrigin>	Herkunft des Konzepts, d.h. Pattern, über das es ins Aufgaben-Modell gelangte	EMPTY	?
<CCPT_OriginPatternID>	Eindeutiger Bezeichner des Musters	#PCDATA	? (1)
<CCPT_OriginPatternVersion>	Versionsnummer des Musters	#PCDATA	? (1)
<CCPT_OriginPatternRevision>	Revisionsnummer des Musters	#PCDATA	? (1)
<CCPT_OriginPatternName>	Bezeichnung des Musters	#PCDATA	?
<CCPT_Preconditions>	Liste von Vorbedingungen, die erfüllt sein müssen, bevor das Konzept verwendet werden kann	EMPTY	?
<CCPT_Precondition>	Einzelne Vorbedingung	EMPTY[264]	* (+)
<CCPT_Postconditions>	Liste von Nachbedingungen, die sich ergeben, nachdem das Konzept verwendet wurde	EMPTY	?
<CCPT_Postcondition>	Einzelne Nachbedingung	EMPTY[264]	* (+)
<CCPT_DataLink>	Verweis auf ein Datenelement im Datenmodell der zugrunde liegenden Anwendung	EMPTY	?
<CCPT_DataLinkID>	Eindeutiger Bezeichner des Verweises	#PCDATA	? (1)
<CCPT_DataLinkName>	Bezeichnung des Verweises	#PCDATA	? (1)
<CCPT_DataLinkDescription>	Texuelle Beschreibung des Verweises	#PCDATA	?
<CCPT_DataLinkReference>	Referenz zum betreffenden Datenelement, z. B. der Name einer Klasse	#PCDATA	? (1)
<CCPT_TaskLinks>	Liste von Verweisen auf Aufgaben im Aufgaben-Modell, die das Konzept benötigen	EMPTY	?
<CCPT_TaskLink>	Einzelner Verweis auf eine Aufgabe	EMPTY	* (+)

[262] (True | False)
[263] (Dummy | Werte sieheTabelle 22)
[264] Weitergehende Beschreibungsstruktur siehe Tabelle 65

Element	Beschreibung	Typ	Kardinalität
<CCPT_TaskID>	Eindeutiger Bezeichner der Aufgabe im zuhörigen Aufgaben-Modell-Fragment	#PCDATA	? (1)
<CCPT_TaskName>	Bezeichnung dieser Aufgabe	#PCDATA	? (1)

G.2.3 Beschreibungselemente eines Dialog-Modell-Fragements

Tabelle 83 BESCHREIBUNGSELEMENTE VON PPSL FÜR DIALOG-MODELLS-FRAGMENTE

Element	Beschreibung	Typ	Kardinalität
<DMF_IncludesDummy>	Kennzeichnung, ob das Dialog-Modell-Fragment einen oder mehrere Dummy-Dialoge zur interaktiven Gestaltung variabler Modellanteile einhält	Enumeration[265]	?
<DMF_ContextModelReferences>	Liste von referenzierten Kontext-Modellen, für die das Dialog-Modell erstellt wurde	EMPTY	?
<DMF_ContextModelReference>	Einzelnes referenziertes Kontext-Modell	EMPTY	* (+)
<DMF_ContextModelID>	Eindeutiger Bezeichner des Kontext-Modells	#PCDATA	? (1)
<DMF_ContextModelVersion>	Versionsnummer des Kontext-Modells	#PCDATA	? (1)
<DMF_ContextModelRevision>	Revisionsnummer des Kontext-Modells	#PCDATA	? (1)
<DMF_ContextModelName>	Bezeichnung des Kontext-Modells	#PCDATA	?
<DMF_Content>	Inhalt des Dialog-Modell-Fragments	EMPTY	?
<Dialog>	Angaben zu einem Dialog	EMPTY	* (+)
<DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	? (1)
<DialogName>	Bezeichnung des Dialogs	#PCDATA	? (1)
<DialogDescription>	Textuelle Beschreibung des Dialogs	#PCDATA	?
<DialogLabel>	Bezeichnung des Dialogs, die ggf. in der Benutzeroberfläche verwendet wird	#PCDATA	?
<DialogType>	Art des Dialogs, z. B. modaler Dialog	Enumeration[266]	? (1)

[265] (True | False)
[266] (Single, Multi, Modal, Complex, Dummy)

Element	Beschreibung	Typ	Kardinalität
<Position>	Position des Dialogs in der Dialog-Abfolge	EMPTY	?
<Predecessors>	Liste von Dialogen, die den Dialog als Nachfolger haben	EMPTY	?
<Predecessor>	Einzelner vorhergehender Dialog	EMPTY	*
<PRED_DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	? (1)
<PRED_DialogName>	Bezeichnung des Dialogs	#PCDATA	?
<PRED_TransitionType>	Art der Transition, d.h. sequenziell oder nebenläufig	Enumeration[267]	? (1)
<PRED_Trigger>	Auslöser für die Dialog-Transition	EMPTY	? (1)
<PRED_ConceptID>	Eindeutiger Bezeichner des auslösenden Konzepts	#PCDATA	? (1)
<PRED_ConceptName>	Bezeichnung des auslösenden Konzepts	#PCDATA	?
<Successors>	Liste von nachfolgenden Dialogen	EMPTY	?
<Successor>	Einzelner nachfolgender Dialog	EMPTY	* (+)
<SUCC_DialogID>	Eindeutiger Bezeichner des Dialogs	#PCDATA	? (1)
<SUCC_DialogName>	Bezeichnung des Dialogs	#PCDATA	?
<SUCC_TransitionType>	Art der Transition, d.h. sequenziell oder nebenläufig	Enumeration[267]	? (1)
<SUCC_Trigger>	Auslöser für die Dialog-Transition	EMPTY	? (1)
<SUCC_ConceptID>	Eindeutiger Bezeichner des auslösenden Konzepts	#PCDATA	? (1)
<SUCC_ConceptName>	Bezeichnung der auslösenden Konzepts	#PCDATA	?
<DLG_Tasks>	Liste der Aufgaben aus dem Aufgaben-Modell, die in dem Dialog enthalten sind	EMPTY	?
<DLG_Task>	Angaben zu einer Aufgabe	EMPTY	* (+)
<DLG_TaskID>	Eindeutiger Bezeichner der Aufgabe im zugehörigen Aufgaben-Modell-Fragment	#PCDATA	? (1)
<DLG_TaskName>	Bezeichnung der Aufgabe	#PCDATA	?
<DLG_Processing>	Angabe, ob nur die Aufgabe selbst oder auch alle, dieser hierarchisch untergeordneten Aufgaben Teil des Dialogs sind	Enumeration[268]	? (1)
<SupplementaryConcepts>	Liste von zusätzlichen UI-Konzepten, für die keine Aufgabe im Task-Mo	EMPTY	?

[267] (Sequential | Concurrent)
[268] (Exclusive | Recursive)

Element	Beschreibung	Typ	Kardinalität
	dell existiert		
<SupplementaryConcept>	Spezifikation eines zusätzlichen UI-Konzepts	EMPTY	* (+)
<SupplementaryConceptID>	Eindeutiger Bezeichner des zusätzlichen UI-Konzepts im Konzept-Modell	#PCDATA	? (1)
<SupplementaryConceptName>	Bezeichnung des zusätzlichen UI-Konzepts	#PCDATA	?

H. Mapping der untersuchten Pattern-Beschreibungssprachen

H.1 Zuordnung der Beschreibungselemente von PLML 1.1 zu PPSL

Tabelle 84 MAPPING DER BESCHREIBUNGSELEMENTE ZWISCHEN PLML 1.1 UND PPSL

PLML 1.1 Element/Attribut	Entsprechung in PPSL
<patternID>	<Head> - <UniquePatternID> - <UPID_PatternID>[269]
<name>	<Head> - <Names> - <Name>[270]
<alias>	<Head> - <Names> - <Aliases>[271]
<illustration>	<Body> - <Practice> - <Examples> - <Example>[272]
<problem>	<Body> - <Theory> - <Problem>[272]
<context>	<Body> - <Theory> - <Context>[272]
<forces>	<Body> - <Theory> - <Forces> - <Force>[271,272]
<solution>	<Body> - <Theory> - <Solution>[272]
<synopsis>	<Head> - <Synopsis>
<diagram>	<Body> - <Theory> - <Diagrams> - <Diagram>[271]
<evidence> <example> <rationale>	Strukturierungs-Element wird in dieser Form nicht benötigt <Body> - <Practice> - <Examples> - <Example>[272] <Body> - <Theory> - <Rationale>
<confidence>	<Body> - <Theory> - <Confidence>
<literature>	<Head> - <References> - <Reference>[272]
<implementation>	<Deployment> - <Implementations> - <Implementation>[272]
<related-patterns>	<Relationship> - <RLTN_Information>
<pattern-link> <type> <patternID> <collection> <label>	<Relationship> - <Relations> - <Relation> - <RLTN_Type> - <RLTN_Reference> - <RLTN_PatternID> - <RLTN_Reference> - <RLTN_PatternCompilationName> - <RLTN_Label>
<management> <author> <credits> <creation-date> <last-modified> <revision-number>	Strukturierungs-Element wird in dieser Form nicht benötigt <Head> - <Contribution> - <Authors> - <Author>[272] <Head> - <Contribution> - <Credits> <Head> - <History> - <CreationDate> <Head> - <History> - <LastModified> <Head> - <UniquePatternID> - <UPID_RevisionNumber>[273,274]

[269] Beim Import von Mustern wird der Inhalt in das Element <Head> - <History> - <Ancestry> - <ANCS_PatternID> übernommen

[270] Beim Import von Mustern wird der Inhalt in das Element <Head> - <History> - <Ancestry> - <ANCS_PatternName> übernommen

[271] Separierung einzelner Inhalte zum Zwecke der getrennten Zugreifbarkeit nach Bedarf

[272] Nutzung der weiteren Unterelemente nach Bedarf

[273] Ggf. Separierung von Revisions- und Versionsnummer (<Head> - <UniquePatternID> - <UPID_VersionNumber>)

[274] Beim Import von Mustern wird der Inhalt in die Elemente <Head> - <History> - <Ancestry> - <ANCS_RevisionNumber> bzw. <ANCS_VersionNumber> übernommen

H.2 Zuordnung der Beschreibungselemente von PLML 1.2 zu PPSL

Tabelle 85 MAPPING DER BESCHREIBUNGSELEMENTE ZWISCHEN PLML 1.2 UND PPSL (ÄNDERUNGEN IM VERGLEICH ZU PLML 1.1)

PLML 1.2 Element/Attribut	Entsprechung in PPSL
<collection>	<Head> - <Classification> - <CLSS_PatternCompilationName>[275]
<forces> <force>	<Body> - <Theory> - <Forces> <Body> - <Theory> - <Forces> - <Force>[276]
<example> <example-name> <example-diagram> <description> <knownuses>	<Body> - <Practice> - <Examples> - <Example> - <XMPL_Label> - <XMPL_Diagram> - <XMPL_Description> <Body> - <Practice> - <KnownUses>[276]
<literature> <workname> <reference>	Strukturierungs-Element wird in dieser Form nicht benötigt <Head> - <References> - <Reference> - <RFNC_Title> <Head> - <References> - <Reference> - <RFNC_Reference>
<implementation> <implementation-name> <code> <otherdetails>	<Deployment> - <Implementations> - <Implementation> - <IMPL_Label> - <IMPL_Code> - <IMPL_Description>
<pattern-link> <type> <patternID> <collection> <label> <revision-number>	<Relationship> - <Relations> - <Relation> - <RLTN_Type> - <RLTN_Reference> - <RLTN_PatternID> - <RLTN_Reference> - <RLTN_PatternCompilationName> - <RLTN_Label> - <RLTN_Reference> - <RLTN_RevisionNumber>[277]
<management> <author> <credits> <creation-date> <last-modified> <revision-number> <change-log> <log-creation-date> <log-content>	Strukturierungs-Element wird in dieser Form nicht benötigt <Head> - <Contribution> - <Authors> - <Author> <Head> - <Contribution> - <Credits> <Head> - <History> - <CreationDate> <Head> - <History> - <LastModified> <Head> - <UniquePatternID> - <UPID_RevisionNumber>[278,279] <Head> - <History> - <Changes> - <Change> - <CHNG_Date>[280] - <Change> - <CHNG_Description>[281]

[275] Beim Import von Mustern wird der Inhalt in das Element <Head> - <History> - <Ancestry> - <ANCS_PatternCompilationName> übernommen

[276] Nutzung der weiteren Unterelemente nach Bedarf

[277] Ggf. Separierung von Revisions- und Versionsnummer (<Relationship> - <Relations> - <Relation> - <RLTN_Reference> - <RLTN_VersionNumber>)

[278] Ggf. Separierung von Revisions- und Versionsnummer (<Head> - <UniquePatternID> - <UPID_VersionNumber>)

[279] Beim Import von Mustern wird der Inhalt in die Elemente <Head> - <History> - <Ancestry> - <ANCS_RevisionNumber> bzw. <ANCS_VersionNumber> übernommen

[280] Das Datum der Erzeugung des Change-Logs entspricht dem Datum des ersten Log-Eintrags

[281] Beim Import von Mustern können die Inhalte auch den anderen zur Verfügung stehenden Unterelementen zugeordnet werden

H.3 Zuordnung der Beschreibungselemente von PLMLx zu PPSL

Tabelle 86 MAPPING DER BESCHREIBUNGSELEMENTE ZWISCHEN PLMLX UND PPSL (ÄNDERUNGEN IM VERGLEICH ZU PLML 1.1)

PLMLx Element/Attribut	Entsprechung in PPSL
<acknowledgements>	<Head> - <Contribution> - <Credits>
<evidence> <example> <rationale>	Strukturierungs-Element wird in dieser Form nicht benötigt <Body> - <Practice> - <Examples> - <Example>[282] <Body> - <Theory> - <Rationale>
<organization> <category> <collection> <classification>	<Head> - <Classification> - <CLSS_Category> - <CLSS_PatternCompilationName> - <CLSS_PatternType>
<resulting-context>	<Body> - <Theory> - <ResultingContext>
<management> <author> <change-log> <change> <author> <description> <version> <date> <majorNo> <minorNo> <copyright> <creation-date> <credits> <last-modified> <license> <license-type> <ulink> <revision-number>	Strukturierungs-Element wird in dieser Form nicht benötigt <Head> - <Contribution> - <Authors> - <Author>[282] <Head> - <History> - <Changes> - <Change> - <CHNG_Originator> - <CHNG_Description> Strukturierungs-Element wird in dieser Form nicht benötigt - <CHNG_Date> - <CHNG_VersionNumber> - <CHNG_RevisionNumber> <Head> - <LegalFoundation> - <Copyright> <Head> - <History> - <CreationDate> <Head> - <Contribution> - <Credits> <Head> - <History> - <LastModified> <Head> - <LegalFoundation> - <License> - <LCNS_Type> - <LCNS_Agreement> <Head> - <UniquePatternID> - <UPID_RevisionNumber>[283,284]

[282] Nutzung der weiteren Unterelemente nach Bedarf

[283] Ggf. Separierung von Revisions- und Versionsnummer (<Head> - <UniquePatternID> - <UPID_VersionNumber>)

[284] Beim Import von Mustern wird der Inhalt in die Elemente <Head> - <History> - <Ancestry> - <ANCS_RevisionNumber> bzw. <ANCS_VersionNumber> übernommen

H.4 Zuordnung der Beschreibungselemente von PCML zu PPSL

Tabelle 87 MAPPING DER BESCHREIBUNGSELEMENTE ZWISCHEN PCML UND PPSL

PCML Element/Attribut	Entsprechung in PPSL
<namespace>	<Head> - <Classification> - <CLSS_PatternCompilationName>
<name>	<Head> - <Names> - <Name>
<abstraction>	<Head> - <Classification> - <CLSS_AbstractionLevel>
<domain>	<Head> - <Classification> - <CLSS_Domain>
<authors> <author> <name> <organization> <description> <URL> <display-name> <address>	<Head> - <Contribution> - <Authors> - <Author> - <ATHR_LastName>[285] - <ATHR_Organization> - <ATHR_PersonalData> - <ATHR_Link> - <ATHR_LinkDisplayName> - <ATHR_URL>
<version> <revision> <date> <description> <release-notes> <copyright> <license> <artifacts>	<Head> - <History> - <Changes> - <Change> - <CHNG_RevisionNumber>[286] - <CHNG_Date> - <CHNG_Description> - <CHNG_Description> <Head> - <LegalFoundation> - <Copyright> <Head> - <LegalFoundation> - <License>[287] - <CHNG_Description>
<akas> <aka>	<Body> - <Practice> - <KnownUses> - <KnownUse>[287]
<keywords> <keyword>	<Head> - <Classification> - <CLSS_Keywords> - <CLSS_Keyword>
<context> <summary> <description>	<Body> - <Theory> - <Context> - <CNTX_Digest> - <CNTX_<Elaboration>
<forces> <force> <summary> <description>	<Body> - <Theory> - <Forces> - <Force> - <FRCE_Digest> - <FRCE_Elaboration>
<problem> <summary> <description>	<Body> - <Theory> - <Problem> - <PRBL_Digest> - <PRBL_Elaboration>

[285] Ggf. Separierung von Nach- und Vornamen (<Head> - <Contribution> - <Authors> - <Author> - <ATHR_FirstName>)

[286] Ggf. Separierung von Revisions- und Versionsnummer (<Head> - <History> - <Changes> - <Change> - <CHNG_VersionNumber>)

[287] Nutzung der weiteren Unterelemente nach Bedarf

PCML Element/Attribut	Entsprechung in PPSL
<solution> <summary> <description> <participants> <name> <required> <description> <stucture> <description> <artifacts> <collaboration> <description> <artifacts>	<Body> - <Theory> - <Solution> - <SLTN_Digest> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration> - <SLTN_Elaboration>
<consequences> <consequence> <summary> <description>	<Body> - <Theory> - <ResultingContext> <Body> - <Theory> - <ResultingContext> <Body> - <Theory> - <ResultingContext> <Body> - <Theory> - <ResultingContext>
<relationships> <relationship> <namespace> <name> <type> <summary> <description>	<Relationships> - <Relations> - <Relation> - <RLTN_Reference> - <RLTN_PatternCompilationName> - <RLTN_Reference> - <RLTN_PatternName> - <RLTN_Type> <RLTN_Description> <RLTN_Description>
<artifacts> <artifact> <name> <type> <description>	<Head> - <References> - <Reference> - <RFNC_Title> - <RFNC_Type> - <RFNC_Description>

H.5 Zuordnung der Beschreibungselemente von TPML zu PPSL

Tabelle 88 MAPPING DER BESCHREIBUNGSELEMENTE ZWISCHEN TPML UND PPSL

TPML Element/Attribut	Entsprechung in PPSL
<Name>	<Head> - <Names> - <Name>
<Problem>	<Body> - <Theory> - <Problem>[288]
<Context>	<Body> - <Theory> - <Context>[288]
<Solution>	<Body> - <Theory> - <Solution>[288]
<Rationale>	<Body> - <Theory> - <Rationale>
<Body> <Task> <ID> <Name> <Type> <Category> <Relation> <Order> <SubTasks> <TaskTemplate> <ID> <VariableDef> <Type> <Name> <Description> <Name> <Text> <Variable> <Type> <Category> <Relation> <Order> <SubTasks>	Strukturierungs-Element wird in dieser Form nicht benötigt <Deployment> - <PaMGIS> - <ModelFragments> - <ModelFrgament> - <MDFR_Fragment> <TaskID> <TaskName> - <TSKN_Prefix> <TaskType> Keine Entsprechung[289] <TemporalOperator> <Position>[290] <SubTasks> <Deployment> - <PaMGIS> - <ModelFragments> - <ModelFragment> - <MDFR_Fragment> <TaskID> <Variables> - <Variable> <VRBL_Type> <VRBL_Name> <VRBL_Annotation> <TaskName> <TSKN_Prefix> <TSKN_VariableName> <TaskType> Keine Entsprechung[289] <TemporalOperator> <Position>[290] <SubTasks>

[288] Nutzung der weiteren Unterelemente nach Bedarf

[289] Die Bedeutung dieses Beschreibungselements konnte der Dokumentation von TPML nicht entnommen werden

[290] Die Information wird in dieser Form nicht benötigt, da bei PPSL die Reihenfolge der Aufgaben über eine Verkettung mithilfe des Beschreibungselements <Position> beschrieben wird

I. Beispielhafte Pattern-Spezifikation

In diesem Anhang befindet sich die Spezifikationen des Musters „Poll“ im PPSL-Format.

Aus Platz- und Übersichtsgründen sind im Wesentlichen jeweils nur die verpflichtenden und die für das generelle Verständnis wichtigen Beschreibungselemente angegeben.

Des Weiteren sind bei allen Zeilen, die über die Seitenbreite hinausgehen, entsprechende Zeilenumbrüche eingefügt und nachfolgende Texteinrückungen vorgenommen worden. Diese gehören zwar nicht zu den eigentlichen Inhalten, erhöhen aber die Lesbarkeit der Spezifikationen im ausgedruckten Format erheblich. Zur besseren Orientierung sind darüber hinaus die Top-Level-Elemente von PPSL jeweils fett dargestellt.

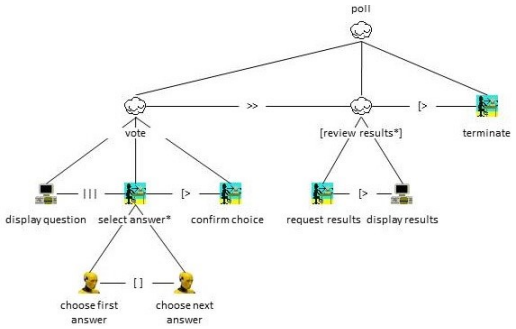
Mit gelber Farbe hinterlegte Stellen in der Pattern-Spezifikation müssen beim Anwenden des Musters durch das Tool *Pattern / Model Selection & Assignment* an den jeweiligen Kontext des Entwicklungsprojekts angepasst werden. Bei den grün markierten Stellen erfolgt die Anpassung mithilfe eines User-Dialogs.

Das Spezifikationsbeispiel ist in englischer Sprache gehalten, da es im Rahmen einer internationalen Publikation verwendet wurde [200].

I.1 „Poll“-Pattern

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0100"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0100_0001"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0000"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Sample"</CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Poll"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>"Pattern in Interaction Design"
    </ANCS_PatternCompilationName>
    <ANCS_PatternName>"Poll"</ANCS_PatternName>
    <ANCS_Link>
     <ANCS_LinkDisplayName>"Patterns in Interaction Design – Poll"
     </ANCS_LinkDisplayName>
     <ANCS_URL>"http://www.welie.com/patterns/showPattern.php?patternID=poll"
     </ANCS_URL>
    </ANCS_Link>
   </Ancestry>
   <CreationDate>"30.01.2017"</CreationDate>
   <LastModified>"30.01.2017"</LastModified>
```

```
 </History>
</Head>
<Body>
 <Theory>
  <Problem>
   <PRBL_Digest>"Ask users for feedback on a specific topic."</PRBL_Digest>
  </Problem>
  <Context>
   <CNTX_Digest>"You are designing a new application where interaction with the users
                 is desired. Typically this will be an interactive application where
                 visitors are to be encouraged to share their opinions and improve
                 interactivity."
   </CNTX_Digest>
  </Context>
  <Solution>
   <SLTN_Digest>"List the statements as exclusive options and optionally present the
                 results after having voted."</SLTN_Digest>
  </Solution>
 </Theory>
 <Practice>
  <KnownUses>
   <KNWN_KnownUseID>""</KNWN_KnownUseID>
   <KNWN_Label>""</KNWN_Label>
   <KNWN_Description>""</KNWN_Description>
   <KNWN_Images>
    <KNWN_Image></KNWN_Image>
   </KNWN_Images>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <Relations>
  <Relation>
   <RLTN_RelationID>""</RLTN_RelationID>
   <RLTN_Label>""</RLTN_Label>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN-Type>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>""</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>""</RLTN_PatternCompilationName>
    <RLTN_PatternID>""</RLTN_PatternID>
    <RLTN_PatternName>""</RLTN_PatternName>
    <RLTN_VersionNumber>""</RLTN_VersionNumber>
    <RLTN_RevisionNumber>""</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0100_01"</MDFR_FragmentID>
    <MDFR_Label>"Poll"</MDFR_Label>
    <MDFR_Purpose>"Request the user's opinion about a specific topic and provide the
                   opportunity to review the overall results of the inquiry"
    </MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>True</IncludesDummy>
 <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0100_01_0001"</TaskID>
   <TaskName>"poll"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0100_0001"</TaskOriginPatternID>
    <TaskOriginPatternName>"Poll"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0000"</TaskOriginPatternRevision>
   </TaskOrigin>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Interleaving</TemporalOperator>
   <UIConcepts></UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0100_01_0002"</TaskID>
     <TaskName>"vote"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Abstraction</TaskType>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>False</Conditional>
     <Preconditions></Preconditions>
     <Postconditions></Postconditions>
     <Position>
      <Parent>
       <ParentID>"__TSK_0100_01_0001"</ParentID>
       <ParentName>"poll"</ParentName>
      </Parent>
      <SiblingLeft>
       <SiblingLeftID>""</SiblingLeftID>
       <SiblingLeftName>""</SiblingLeftName>
      </SiblingLeft>
```

```
 <SiblingRight>
  <SiblingRightID>"__TSK_0100_01_0003"<SiblingRightID>
  <SiblingRightName>"review results"</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>SequentialEnabling</TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0100_01_0005"</TaskID>
  <TaskName>"display question"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Application</TaskType>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False"</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0100_01_0002"</ParentID>
    <ParentName>"vote"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0100_01_0006"<SiblingRightID>
    <SiblingRightName>"select answer"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0100_01_0001"</UIConceptID>
    <UIConceptName>"Question"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0100_01_0006"</TaskID>
  <TaskName>"select answer"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>"Interaction"</TaskType>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>True</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0100_01_0002"</ParentID>
    <ParentName>"vote"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>"__TSK_0100_01_0005"</SiblingLeftID>
    <SiblingLeftName>"display question"</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0100_01_0007"<SiblingRightID>
    <SiblingRightName>"confirm choice"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Disabling</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0100_01_0002"</UIConceptID>
    <UIConceptName>"Answer"</UIConceptName>
```

```
 </UIConcept
</UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0100_01_0008"</TaskID>
  <TaskName>"choose first answer"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Dummy</TaskType>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0100_01_0006"</ParentID>
    <ParentName>"select answer"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"  TSK_0100_01_0009"<SiblingRightID>
    <SiblingRightName>"choose next answer"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Choice</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"  CPT_0100_01_0007"</UIConceptID>
    <UIConceptName>"Answer 1"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"  TSK_0100_01_0009"</TaskID>
  <TaskName>"choose next answer"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Dummy</TaskType>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0100_01_0006"</ParentID>
    <ParentName>"select answer"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>"  TSK_0100_01_0008"</SiblingLeftID>
    <SiblingLeftName>"choose first answer"</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>""<SiblingRightID>
    <SiblingRightName>""</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Choice</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"  CPT_0100_01_0008"</UIConceptID>
    <UIConceptName>"Answer 2"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
</Subtasks>
</Subtask>
```

```
  <Subtask>
   <TaskID>"__TSK_0100_01_0007"</TaskID>
   <TaskName>"confirm choice"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Interaction</TaskType>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>"__TSK_0100_01_0002"</ParentID>
     <ParentName>"vote"</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>"__TSK_0100_01_0006"</SiblingLeftID>
     <SiblingLeftName>"select answer"</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator></TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"__CPT_0100_01_0003"</UIConceptID>
     <UIConceptName>"Confirmation"</UIConceptName>
    </UIConcept
   </UIConcepts>
  </Subtask>
 </Subtasks>
</Subtask
<Subtask>
 <TaskID>"__TSK_0100_01_0003"</TaskID>
 <TaskName>"review results"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>True</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0100_01_0001"</ParentID>
   <ParentName>"poll"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0100_01_0002"</SiblingLeftID>
   <SiblingLeftName>"vote"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0100_01_0004"<SiblingRightID>
   <SiblingRightName>"terminate"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Disabling</TemporalOperator>
 <UIConcepts></UIConcepts>
 <Subtasks>
  <Subtask>
   <TaskID>"__TSK_0100_01_0010"</TaskID>
   <TaskName>"request results"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>"Interaction"</TaskType>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
```

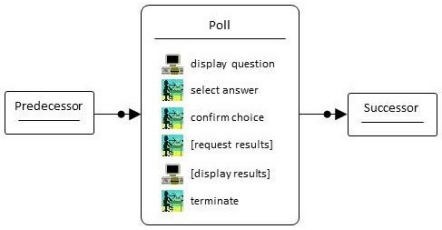
```
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>"__TSK_0100_01_0003"</ParentID>
     <ParentName>"review results"</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>"__TSK_0100_01_0011"<SiblingRightID>
     <SiblingRightName>"display results"</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Disabling"/TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"__CPT_0100_01_0004"</UIConceptID>
     <UIConceptName>"Request"</UIConceptName>
    </UIConcept
   </UIConcepts>
  </Subtask>
  <Subtask>
   <TaskID>"__TSK_0100_01_0011"</TaskID>
   <TaskName>"display results"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Application</TaskType>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>"__TSK_0100_01_0003"</ParentID>
     <ParentName>"review results"</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>"__TSK_0100_01_0010"</SiblingLeftID>
     <SiblingLeftName>"request results"</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator></TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"__CPT_0100_01_0005"</UIConceptID>
     <UIConceptName>"Results"</UIConceptName>
    </UIConcept
   </UIConcepts>
  </Subtask>
 </Subtasks>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0100_01_0004"</TaskID>
 <TaskName>"teminate"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>"Interaction"</TaskType>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
```

```
      <Position>
       <Parent>
        <ParentID>"__TSK_0100_01_0001"</ParentID>
        <ParentName>"poll"</ParentName>
       </Parent>
       <SiblingLeft>
        <SiblingLeftID>"__TSK_0100_01_0003"</SiblingLeftID>
        <SiblingLeftName>"review results"</SiblingLeftName>
       </SiblingLeft>
       <SiblingRight>
        <SiblingRightID>""<SiblingRightID>
        <SiblingRightName>""</SiblingRightName>
       </SiblingRight>
      </Position>
      <TemporalOperator></TemporalOperator>
      <UIConcepts>
       <UIConcept>
        <UIConceptID>"__CPT_0100_01_0006"</UIConceptID>
        <UIConceptName>"Termination"</UIConceptName>
       </UIConcept
      </UIConcepts>
      </Subtasks>
     </Subtask>
    </Subtasks>
   </Subtask>
  </TMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0100_01"</MDFR_FragmentID>
 <MDFR_Label>"Concepts for poll pattern"</MDFR_Label>
 <MDFR_Purpose>"Specification of all concepts relevant for poll pattern"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram></MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>True</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0100_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Question"</CCPT_ConceptName>
    <CCPT_Description>"Question to be answered by the user"</CCPT_Description>
    <CCPT_Label>""</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>DataOutput</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0100_01_0005"</CCPT_TaskID>
      <CCPT_TaskName>"display question"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0100_01_0002"</CCPT_ConceptID>
    <CCPT_ConceptName>"Answer"</CCPT_ConceptName>
    <CCPT_Description>"Answer chosen by the user"</CCPT_Description>
    <CCPT_Label>""</CCPT_Label>
```

```
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"Integer"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0100_01_0006"</CCPT_TaskID>
   <CCPT_TaskName>"select answer"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0100_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Confirmation"</CCPT_ConceptName>
 <CCPT_Description>"Confirmation of chosen answer"</CCPT_Description>
 <CCPT_Label>"confirm choice"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>Activator</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0100_01_0007"</CCPT_TaskID>
   <CCPT_TaskName>"confirm choice"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0100_01_0004"</CCPT_ConceptID>
 <CCPT_ConceptName>"Request"</CCPT_ConceptName>
 <CCPT_Description>"Request overall results to be displayed"</CCPT_Description>
 <CCPT_Label>"request results"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Activator</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
```

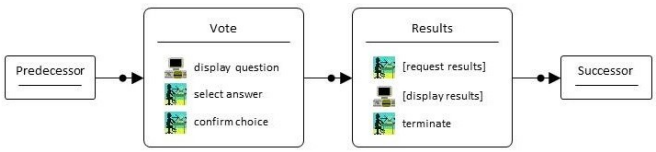
```
   <CCPT_TaskID>"__TSK_0100_01_0010"</CCPT_TaskID>
   <CCPT_TaskName>"request results"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0100_01_0005"</CCPT_ConceptID>
 <CCPT_ConceptName>"Results"</CCPT_ConceptName>
 <CCPT_Description>"Overall results tob e displayed"</CCPT_Description>
 <CCPT_Label>""</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>DataOutput</CCPT_ConceptType>
 <CCPT_DataType>"Blob"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0100_01_0011"</CCPT_TaskID>
   <CCPT_TaskName>"display results"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0100_01_0006"</CCPT_ConceptID>
 <CCPT_ConceptName>"Termination"</CCPT_ConceptName>
 <CCPT_Description>"terminate poll"</CCPT_Description>
 <CCPT_Label>"abandon"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Navigator</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0100_01_0004"</CCPT_TaskID>
   <CCPT_TaskName>"terminate"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0100_01_0007"</CCPT_ConceptID>
 <CCPT_ConceptName>"Answer 1"</CCPT_ConceptName>
 <CCPT_Description>"Erste Antwortmöglichkeit"</CCPT_Description>
 <CCPT_Label>""</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
```

```
      <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
      <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
      <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
      <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
     </CCPT_ConceptOrigin>
     <CCPT_Preconditions></CCPT_Preconditions>
     <CCPT_Postconditions><CCPT_Postconditions>
     <CCPT_DataLink></CCPT_DataLink>
     <CCPT_TaskLinks>
      <CCPT_TaskLink>
       <CCPT_TaskID>"__TSK_0100_01_0008"</CCPT_TaskID>
       <CCPT_TaskName>"choose first answer"</CCPT_TaskName>
      </CCPT_TaskLink>
     </CCPT_TaskLinks>
    </Concept>
    <Concept>
     <CCPT_ConceptID>"__CPT_0100_01_0008"</CCPT_ConceptID>
     <CCPT_ConceptName>"Answer 2"</CCPT_ConceptName>
     <CCPT_Description>"Weitere Antwortmöglichkeit"</CCPT_Description>
     <CCPT_Label>""</CCPT_Label>
     <CCPT_Perceptible>True</CCPT_Perceptible>
     <CCPT_Enabled>True</CCPT_Enabled>
     <CCPT_Required>False</CCPT_Required>
     <CCPT_ConceptType>Dummy</CCPT_ConceptType>
     <CCPT_DataType>"Integer"</CCPT_DataType>
     <CCPT_Value>""</CCPT_Value>
     <CCPT_ConceptOrigin>
      <CCPT_OriginPatternID>"PPT_0100_0001"</CCPT_OriginPatternID>
      <CCPT_OriginPatternName>"Poll"</CCPT_OriginPatternName>
      <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
      <CCPT_OriginPatternRevision>"0000"</CCPT_OriginPatternRevision>
     </CCPT_ConceptOrigin>
     <CCPT_Preconditions></CCPT_Preconditions>
     <CCPT_Postconditions><CCPT_Postconditions>
     <CCPT_DataLink></CCPT_DataLink>
     <CCPT_TaskLinks>
      <CCPT_TaskLink>
       <CCPT_TaskID>"__TSK_0100_01_0009"</CCPT_TaskID>
       <CCPT_TaskName>"choose next answer"</CCPT_TaskName>
      </CCPT_TaskLink>
     </CCPT_TaskLinks>
    </Concept>
   </CMF_Content>
  </MDFR_Fragment>
 </ModelFragment>
 <ModelFragment>
  <MDFR_Type>Dialog</MDFR_Type>
  <MDFR_FragmentID>"__DMF_0100_01"</MDFR_FragmentID>
  <MDFR_Label>"Poll"</MDFR_Label>
  <MDFR_Purpose>"Dialog Model Fragment for large screens"</MDFR_Purpose>
  <MDFR_Annotation>""</MDFR_Annotation>
  <MDFR_Diagram>
```




```
  </MDFR_Diagram>
  <MDFR_Fragment>
   <DMF_IncludesDummy>False</DMF_IncludesDummy>
   <DMF_ContextModelReferences></DMF_ContextModelReferences>
   <DMF_Content>
    <Dialog>
     <DialogID>"__DLG_0100_01_01"</DialogID>
     <DialogName>"Poll"</DialogName>
     <DialogDescription>"Poll for large screens"</DialogDescription>
     <DialogLabel>"Poll"</DialogLabel>
```

```
    <DialogType>Modal</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED_DialogID>""</PRED_DialogID>
       <PRED_DialogName>""</PRED_DialogName>
       <PRED_TransitionType>Sequential</PRED_TransitionType>
       <PRED_Trigger>
        <PRED_ConceptID>""</PRED_ConceptID>
        <PRED_ConceptName>""</PRED_ConceptName>
       </PRED_Trigger>
      </Predecessor>
     <Predecessors>
     <Successors>
      <Successors>
       <SUCC_DialogID>""</SUCC_DialogID>
       <SUCC_DialogName>""</SUCC_DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>"__CPT_0100_01_0006"</SUCC_ConceptID>
        <SUCC_ConceptName>"Termination"</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0100_01_0002"</DLG_TaskID>
      <DLG_TaskName>"vote"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     <DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0100_01_0003"</DLG_TaskID>
      <DLG_TaskName>"review results"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     <DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0100_01_0004"</DLG_TaskID>
      <DLG_TaskName>"terminate"</DLG_TaskName>
      <DLG_Processing>Exclusive</DLG_Processing>
     </DLF_Task>
    </DLG_Tasks>
   </Dialog>
  </DMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0100_02"</MDFR_FragmentID>
 <MDFR_Label>"Poll"</MDFR_Label>
 <MDFR_Purpose>"Dialog Model Fragment for small screens"</MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```




```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_ContextModelReferences></DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0100_02_01"</DialogID>
    <DialogName>"Vote"</DialogName>
    <DialogDescription>"Vote part for small screens"</DialogDescription>
    <DialogLabel>"Vote"</DialogLabel>
    <DialogType>Modal</DialogType>
    <Position>
     <Predecessors>
```

```
    <Predecessor>
     <PRED_DialogID>""</PRED_DialogID>
     <PRED_DialogName>""</PRED_DialogName>
     <PRED_TransitionType>Sequential</PRED_TransitionType>
     <PRED_Trigger>
      <PRED_ConceptID>""</PRED_ConceptID>
      <PRED_ConceptName>""</PRED_ConceptName>
     </PRED_Trigger>
    </Predecessor>
   <Predecessors>
   <Successors>
    <Successor>
     <SUCC_DialogID>"__DLG_0100_02_02"</SUCC_DialogID>
     <SUCC_DialogName>"Results"</SUCC_DialogName>
     <SUCC_TransitionType>Sequential</SUCC_TransitionType>
     <SUCC_Trigger>
      <SUCC_ConceptID>"__CPT_0100_01_0003"</SUCC_ConceptID>
      <SUCC_ConceptName>"Confirmation"</SUCC_ConceptName>
     </SUCC_Trigger>
    </Successor>
   </Successors>
  </Position>
  <DLG_Tasks>
   <DLG_Task>
    <DLG_TaskID>"__TSK_0100_01_0002"</DLG_TaskID>
    <DLG_TaskName>"vote"</DLG_TaskName>
    <DLG_Processing>Recursive</DLG_Processing>
   <DLG_Task>
  </DLG_Tasks>
</Dialog>
<Dialog>
 <DialogID>"__DLG_0100_02_02"</DialogID>
 <DialogName>"Results"</DialogName>
 <DialogDescription>"Results part for small screens"</DialogDescription>
 <DialogLabel>"Results"</DialogLabel>
 <DialogType>Modal</DialogType>
 <Position>
  <Predecessors>
   <Predecessor>
    <PRED_DialogID>"__DLG_0100_02_01"</PRED_DialogID>
    <PRED_DialogName>"Vote"</PRED_DialogName>
    <PRED_TransitionType>Sequential</PRED_TransitionType>
    <PRED_Trigger>
     <PRED_ConceptID>""</PRED_ConceptID>
     <PRED_ConceptName>""</PRED_ConceptName>
    </PRED_Trigger>
   </Predecessor>
  <Predecessors>
  <Successors>
   <Successor>
    <SUCC_DialogID>""</SUCC_DialogID>
    <SUCC_DialogName>""</SUCC_DialogName>
    <SUCC_TransitionType>Sequential</SUCC_TransitionType>
    <SUCC_Trigger>
     <SUCC_ConceptID>"__CPT_0100_01_0006"</SUCC_ConceptID>
     <SUCC_ConceptName>"Termination"</SUCC_ConceptName>
    </SUCC_Trigger>
   </Successors>
  </Successor>
 </Position>
 <DLG_Tasks>
  <DLG_Task>
   <DLG_TaskID>"__TSK_0100_01_0003"</DLG_TaskID>
   <DLG_TaskName>"review results"</DLG_TaskName>
   <DLG_Processing>Recursive</DLG_Processing>
  <DLG_Task>
  <DLG_Task>
   <DLG_TaskID>"__TSK_0100_01_0004"</DLG_TaskID>
   <DLG_TaskName>"terminate"</DLG_TaskName>
   <DLG_Processing>Exclusive</DLG_Processing>
  <DLG_Task>
```

```
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J. Pattern-Sprache zur Fallstudie

In den folgenden Unterkapiteln befinden sich die Spezifikationen der Muster der Pattern-Sprache für die Domäne „Ticket-Verkauf für die Personenbeförderung“ im PPSL-Format.

Aus Platz- und Übersichtsgründen sind im Wesentlichen jeweils nur die verpflichtenden und die für das Verständnis der Fallstudie benötigten Beschreibungselemente angegeben.

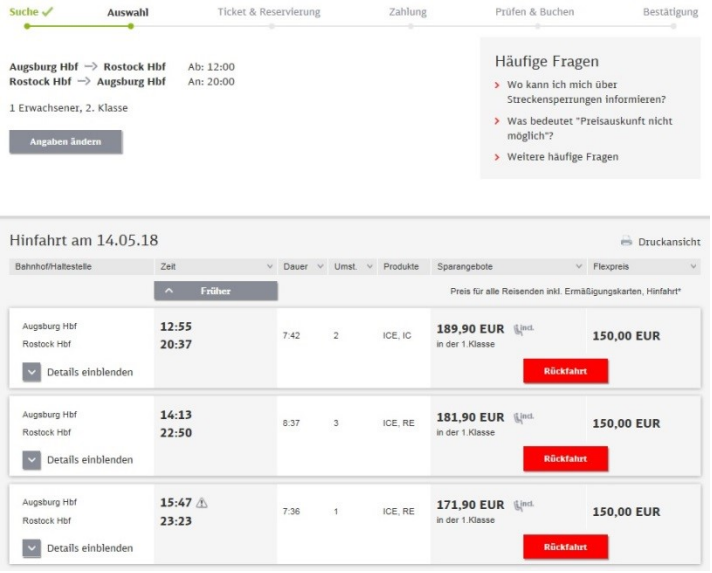
Des Weiteren sind bei allen Zeilen, die über die Seitenbreite hinausgehen, entsprechende Zeilenumbrüche eingefügt und nachfolgende Texteinrückungen vorgenommen worden. Diese gehören zwar nicht zu den eigentlichen Inhalten, erhöhen aber die Lesbarkeit der Spezifikationen im ausgedruckten Format erheblich. Zur besseren Orientierung sind darüber hinaus die Top-Level-Elemente von PPSL jeweils fett dargestellt.

Mit gelber Farbe hinterlegte Stellen in den Pattern-Spezifikationen müssen beim Anwenden des jeweiligen Patterns durch das Tool *Pattern / Model Selection & Assignment* an den jeweiligen Kontext des Entwicklungsprojekts angepasst werden. Bei den grün markierten Stellen erfolgt die Anpassung mithilfe eines User-Dialogs.

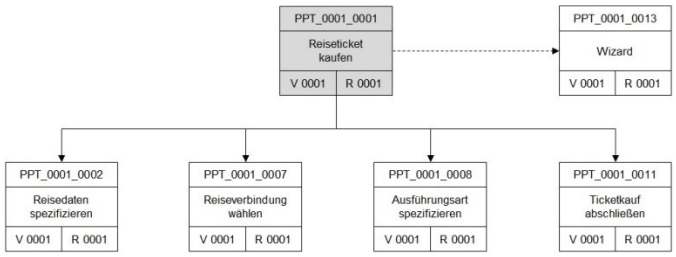
J.1 Reiseticket kaufen

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0001"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reiseticket kaufen"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"14.05.2018"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0001_0001"</CHNG_ChangeID>
     <CHNG_Date>"14.05.2018"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
```

```
    <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
    <CHNG_Link></CHNG_Link>
   </Change>
  </Changes>
 </History>
</Head>
<Body>
 <Theory>
  <Problem>
   <PRBL_Digest>"Es sind die für einen Reiseticketkauf erforderlichen Schritte
                festzulegen."</PRBL_Digest>
  </Problem>
  <Context>
   <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                Flugzeug."</CNTX_Digest>
  </Context>
  <Solution>
   <SLTN_Digest>"Der Ticketkauf erfolgt durch folgende Arbeitsschritte:
                1. Angabe der Reisedaten, anhand derer entsprechende Reiseverbin-
                   dungen gesucht werden können.
                2. Auswahl der gewünschten Reisverbindung
                3. Spezifizieren der Ausführungsart, d. h. Auswahl der angebotenen
                   Zahlungs- und Versandmodalitäten.
                4. Abschließen des Ticketkaufs durch Überprüfen der gemachten
                   Angaben und verbindliche Kaufzusage."</SLTN_Digest>
  </Solution>
 </Theory>
 <Practice>
  <KnownUses>
   <KnownUse>
    <KNWN_KnownUseID>"0001_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Ticketkauf - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Ablauf des Ticketkaufs im Online-Portal der Deutschen Bahn
                       (www.bahn.de)"
    </KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```

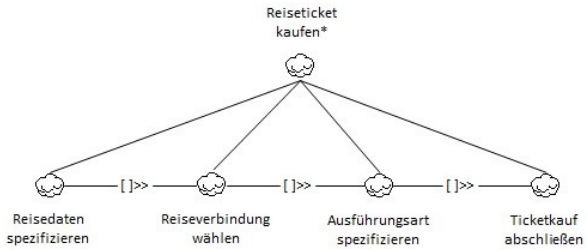



```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
</RLTN_Diagram>
<Relations>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0001_0001"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
   <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0001_0002"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0007"</RLTN_PatternID>
   <RLTN_PatternName>"Reiseverbindung wählen"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0001_0003"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0008"</RLTN_PatternID>
   <RLTN_PatternName>"Ausführungsart spezifizieren"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0001_0004"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0011"</RLTN_PatternID>
   <RLTN_PatternName>"Ticketkauf abschließen"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
```

```
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0001_0005"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Assoziation</RLTN_Type>
   <RLTN_Sense>"nutzt"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0013"</RLTN_PatternID>
    <RLTN_PatternName>"Wizard"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0001_01"</MDFR_FragmentID>
    <MDFR_Label>"Reiseticket kaufen"</MDFR_Label>
    <MDFR_Purpose>"Durchführung eines Ticketkaufs"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>False</IncludesDummy>
     <TMF_ProposedTempOp>SequentialEnabling</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0001_01_0001"</TaskID>
       <TaskName>"Reiseticket kaufen"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Abstraction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0001"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reiseticket kaufen"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>True</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>""</ParentID>
         <ParentName>""</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>""</SiblingLeftID>
         <SiblingLeftName>""</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
```

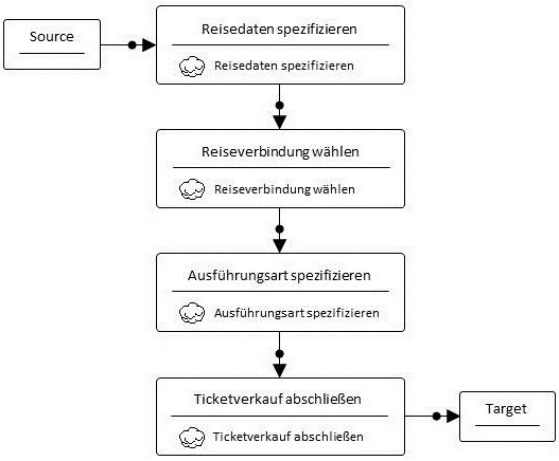
```
  <SiblingRightName>""</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>Disabling</TemporalOperator>
<UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0001_01_0001"</UIConceptID>
    <UIConceptName>"Ticketkauf starten"</UIConceptName>
   </UIConcept>
  </UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0001_01_0002"</TaskID>
  <TaskName>"Reisedaten spezifizieren"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0001"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reiseticket kaufen"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0001_01_0001"</ParentID>
    <ParentName>"Reiseticket kaufen"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0001_01_0003"<SiblingRightID>
    <SiblingRightName>"Reiseverbindung wählen"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>SequentialEnablingInfo</TemporalOperator>
  <UIConcepts></UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0001_01_0003"</TaskID>
  <TaskName>"Reiseverbindung wählen"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0001"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reiseticket kaufen"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0001_01_0001"</ParentID>
    <ParentName>"Reiseticket kaufen"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>"__TSK_0001_01_0002"</SiblingLeftID>
    <SiblingLeftName>"Reisedaten spezifizieren"</SiblingLeftName>
   </SiblingLeft>
```

```
  <SiblingRight>
   <SiblingRightID>"__TSK_0001_01_0004"<SiblingRightID>
   <SiblingRightName>"Ausführungsart spezifizieren"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>SequentialEnablingInfo</TemporalOperator>
 <UIConcepts></UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0001_01_0004"</TaskID>
 <TaskName>"Ausführungsart spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0001"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reiseticket kaufen"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0001_01_0001"</ParentID>
   <ParentName>"Reiseticket kaufen"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0001_01_0003"</SiblingLeftID>
   <SiblingLeftName>"Reiseverbindung wählen"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0001_01_0005"<SiblingRightID>
   <SiblingRightName>"Ticketkauf abschließen"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>SequentialEnablingInfo</TemporalOperator>
 <UIConcepts></UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0001_01_0005"</TaskID>
 <TaskName>"Ticketkauf abschließen"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0001"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reiseticket kaufen"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0001_01_0001"</ParentID>
   <ParentName>"Reiseticket kaufen"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0001_01_0004"</SiblingLeftID>
   <SiblingLeftName>"Ausführungsart spezifizieren"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>""<SiblingRightID>
   <SiblingRightName>" "</SiblingRightName>
```

```
      </SiblingRight>
     </Position>
     <TemporalOperator></TemporalOperator>
     <UIConcepts></UIConcepts>
    </Subtask>
   </Subtasks>
  </Subtask>
 </TMF_Content>
</MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0001_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Reisedaten spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reiseticket kaufen“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0001_01_0001	Ticketkauf starten	Navigator

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>False</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0001_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Ticketkauf starten"</CCPT_ConceptName>
    <CCPT_Description>"Beginnen des Ticketkaufs"</CCPT_Description>
    <CCPT_Label>"Ticketkauf"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>Navigator</CCPT_ConceptType>
    <CCPT_DataType>""</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0001"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reiseticket kaufen"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0001_01_0001"</CCPT_TaskID>
      <CCPT_TaskName>"Reiseticket kaufen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0001_01"</MDFR_FragmentID>
 <MDFR_Label>"Reiseticket kaufen"</MDFR_Label>
 <MDFR_Purpose>"Kauf von Reisetickets - grundlegender Ablauf"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```




```
</MDFR_Diagram>
<MDFR_Fragment>
 <DMF_IncludesDummy>False</DMF_IncludesDummy>
 <DMF_ContextModelReferences>
  <DMF_ContextModelReference>
   <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
   <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
   <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
   <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
  </DMF_ContextModelReference>
  <DMF_ContextModelReference>
   <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
   <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
   <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
   <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
  </DMF_ContextModelReference>
 </DMF_ContextModelReferences>
 <DMF_Content>
  <Dialog>
   <DialogID>"__DLG_0001_01_0001"</DialogID>
   <DialogName>"Reisedaten spezifizieren"</DialogName>
   <DialogDescription>""</DialogDescription>
   <DialogLabel>"Reisedaten spezifizieren"</DialogLabel>
   <DialogType>Complex</DialogType>
   <Position>
    <Predecessors>
     <Predecessor>
      <PRED_DialogID>""</PRED_DialogID>
      <PRED_DialogName>""</PRED_DialogName>
     </Predecessor>
    </Predecessors>
    <Successors>
     <Successor>
      <SUCC_DialogID>"__DLG_0001_01_0002"</SUCC_DialogID>
      <SUCC_DialogName>"Reiseverbindung wählen"</SUCC_DialogName>
      <SUCC_TransitionType>Sequential</SUCC_TransitionType>
      <SUCC_Trigger>
       <SUCC_ConceptID>""</SUCC_ConceptID>
       <SUCC_ConceptName>""</SUCC_ConceptName>
      </SUCC_Trigger>
     </Successor>
    </Successors>
   </Position>
   <DLG_Tasks>
    <DLG_Task>
     <DLG_TaskID>"__TSK_0001_01_0002"</DLG_TaskID>
     <DLG_TaskName>"Reisedaten spezifizieren"</DLG_TaskName>
     <DLG_Processing>Recursive</DLG_Processing>
    </DLG_Task>
   </DLG_Tasks>
  </Dialog>
  <Dialog>
```

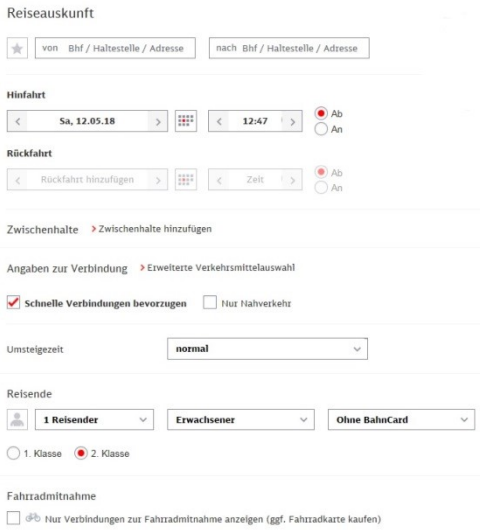
```
 <DialogID>"__DLG_0001_01_0002"</DialogID>
 <DialogName>"Reiseverbindung wählen"</DialogName>
 <DialogDescription>""</DialogDescription>
 <DialogLabel>"Reiseverbindung wählen"</DialogLabel>
 <DialogType>Complex</DialogType>
 <Position>
  <Predecessors>
   <Predecessor>
    <PRED_DialogID>"__DLG_0001_01_0001"</PRED_DialogID>
    <PRED_DialogName>"Reisedaten spezifizieren"</PRED_DialogName>
   </Predecessor>
  </Predecessors>
  <Successors>
   <Successor>
    <SUCC_DialogID>"__DLG_0001_01_0003"</SUCC_DialogID>
    <SUCC_DialogName>"Ausführungsart spezifizieren"</SUCC_DialogName>
    <SUCC_TransitionType>Sequential</SUCC_TransitionType>
    <SUCC_Trigger>
     <SUCC_ConceptID>""</SUCC_ConceptID>
     <SUCC_ConceptName>""</SUCC_ConceptName>
    </SUCC_Trigger>
   </Successor>
  </Successors>
 </Position>
 <DLG_Tasks>
  <DLG_Task>
   <DLG_TaskID>"__TSK_0001_01_0003"</DLG_TaskID>
   <DLG_TaskName>"Reiseverbindung wählen"</DLG_TaskName>
   <DLG_Processing>Recursive</DLG_Processing>
  </DLG_Task>
 </DLG_Tasks>
</Dialog>
<Dialog>
 <DialogID>"__DLG_0001_01_0003"</DialogID>
 <DialogName>"Ausführungsart spezifizieren"</DialogName>
 <DialogDescription>""</DialogDescription>
 <DialogLabel>"Ausführungsart spezifizieren"</DialogLabel>
 <DialogType>Complex</DialogType>
 <Position>
  <Predecessors>
   <Predecessor>
    <PRED_DialogID>"__DLG_0001_01_0002"</PRED_DialogID>
    <PRED_DialogName>"Reiseverbindung wählen"</PRED_DialogName>
   </Predecessor>
  </Predecessors>
  <Successors>
   <Successor>
    <SUCC_DialogID>"__DLG_0001_01_0004"</SUCC_DialogID>
    <SUCC_DialogName>"Ticketverkauf abschließen"</SUCC_DialogName>
    <SUCC_TransitionType>Sequential</SUCC_TransitionType>
    <SUCC_Trigger>
     <SUCC_ConceptID>""</SUCC_ConceptID>
     <SUCC_ConceptName>""</SUCC_ConceptName>
    </SUCC_Trigger>
   </Successor>
  </Successors>
 </Position>
 <DLG_Tasks>
  <DLG_Task>
   <DLG_TaskID>"__TSK_0001_01_0004"</DLG_TaskID>
   <DLG_TaskName>"Ausführungsart spezifizieren"</DLG_TaskName>
   <DLG_Processing>Recursive</DLG_Processing>
  </DLG_Task>
 </DLG_Tasks>
</Dialog>
<Dialog>
 <DialogID>"__DLG_0001_01_0004"</DialogID>
 <DialogName>"Ticketverkauf abschließen"</DialogName>
 <DialogDescription>""</DialogDescription>
 <DialogLabel>"Ticketverkauf abschließen"</DialogLabel>
 <DialogType>Complex</DialogType>
```

```
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>"__DLG_0001_01_0003"</PRED_DialogID>
           <PRED_DialogName>"Ausführungsart spezifizieren"</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>""</SUCC_ConceptID>
            <SUCC_ConceptName>""</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0001_01_0005"</DLG_TaskID>
          <DLG_TaskName>"Ticketverkauf abschließen"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

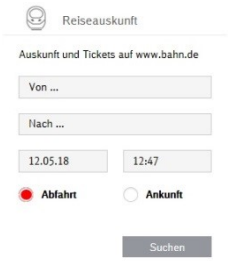
J.2 Reisedaten spezifizieren

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0002"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reisedaten spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"14.05.2018"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0002_0001"</CHNG_ChangeID>
     <CHNG_Date>"14.05.2018"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind diejenigen Reisedaten festzulegen, die für die Suche nach
                  passenden Reiseverbindungen herangezogen werden können."
    </PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zur Reise bzw. Arbeitsschritte werden benötigt:
                  1. Reisestrecke, z. B. durch Angebe von Start- und Zielort
                  2. Reisezeit, z. B. durch Angabe von Datum und Uhrzeit
                  3. Reisende, z. B. Anzahl der Reisenden
                  4. Reiseoptionen, z. B. Angabe der Reiseklasse
                  5. Abschließen der Eingabe und Anfordern der Suchergebnisse"
    </SLTN_Digest>
   </Solution>
```

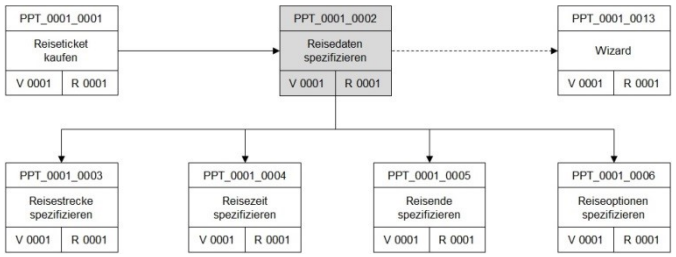
```
</Theory>
<Practice>
 <KnownUses>
  <KnownUse>
   <KNWN_KnownUseID>"0002_0001"</KNWN_KnownUseID>
   <KNWN_Label>"Reisedaten - Deutsche Bahn AG"</KNWN_Label>
   <KNWN_Description>"Angaben der Reisedaten im Online-Portal der Deutschen Bahn
                      (www.bahn.de)"</KNWN_Description>
   <KNWN_Images>
    <KNWN_Image>
```




```
    </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
   <KnownUse>
    <KNWN_KnownUseID>"0002_0002"</KNWN_KnownUseID>
    <KNWN_Label>"Reisedaten - Deutsche Bahn"</KNWN_Label>
    <KNWN_Description>"Angaben der Reisedaten im internationalen Online-Portal der
                       Deutschen Bahn (www.deutsche-bahn.com/de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```

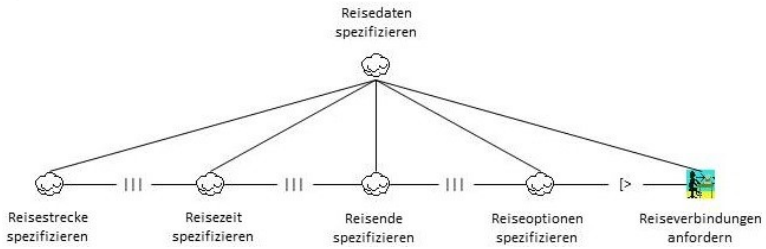



```
    </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
</RLTN_Diagram>
<Relations>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0002_0001"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"ist Teil von"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
   <RLTN_PatternName>"Reiseticket kaufen"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0002_0002"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0003"</RLTN_PatternID>
   <RLTN_PatternName>"Reisestrecke spezifizieren"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0002_0003"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0004"</RLTN_PatternID>
   <RLTN_PatternName>"Reisezeit spezifizieren"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
 </Relation>
 <Relation>
  <RLTN_RelationID>"RLN_0001_0002_0004"</RLTN_RelationID>
  <RLTN_Nature>Internal</RLTN_Nature>
  <RLTN_Type>Komposition</RLTN_Type>
  <RLTN_Sense>"besteht aus"</RLTN_Sense>
  <RLTN_Reference>
   <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
   <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </RLTN_PatternCompilationName>
   <RLTN_PatternID>"PPT_0001_0005"</RLTN_PatternID>
   <RLTN_PatternName>"Reisende spezifizieren"</RLTN_PatternName>
   <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
   <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
  </RLTN_Reference>
```

```
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0002_0005"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"besteht aus"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0006"</RLTN_PatternID>
    <RLTN_PatternName>"Reiseoptionen spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0002_0006"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Assoziation</RLTN_Type>
   <RLTN_Sense>"nutzt"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0013"</RLTN_PatternID>
    <RLTN_PatternName>"Wizard"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0002_01"</MDFR_FragmentID>
    <MDFR_Label>"Reisedaten spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren aller relevanten Parameter zur Auswahl von Reise-
                   verbindungen"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>False</IncludesDummy>
     <TMF_ProposedTempOp>SequentialEnabling</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0002_01_0001"</TaskID>
       <TaskName>"Reisedaten spezifizieren"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Abstraction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
```

```
<Optional>False</Optional>
<Iterative>False</Iterative>
<Conditional>False</Conditional>
<Preconditions></Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
  <ParentID>""</ParentID>
  <ParentName>""</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>""</SiblingLeftID>
  <SiblingLeftName>""</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>""<SiblingRightID>
  <SiblingRightName>""</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>SequentialEnablingInfo</TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0002_01_0002"</TaskID>
  <TaskName>"Reisestrecke spezifizieren"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0002_01_0001"</ParentID>
    <ParentName>"Reisedaten spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0002_01_0003"<SiblingRightID>
    <SiblingRightName>"Reisezeit spezifizieren"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving</TemporalOperator>
  <UIConcepts></UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0002_01_0003"</TaskID>
  <TaskName>"Reisezeit spezifizieren"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
```

```
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0002_01_0001"</ParentID>
   <ParentName>"Reisedaten spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0002_01_0002"</SiblingLeftID>
   <SiblingLeftName>"Reisestrecke spezifizieren"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0002_01_0004"<SiblingRightID>
   <SiblingRightName>"Reisende spezifizieren"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts></UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0002_01_0004"</TaskID>
 <TaskName>"Reisende spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0002_01_0001"</ParentID>
   <ParentName>"Reisedaten spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0002_01_0003"</SiblingLeftID>
   <SiblingLeftName>"Reisezeit spezifizieren"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0002_01_0005"<SiblingRightID>
   <SiblingRightName>"Reiseoptionen spezifizieren"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Disabling</TemporalOperator>
 <UIConcepts></UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0002_01_0005"</TaskID>
 <TaskName>"Reiseoptionen spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
```

```
        <Parent>
         <ParentID>"__TSK_0002_01_0001"</ParentID>
         <ParentName>"Reisedaten spezifizieren"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0002_01_0004"</SiblingLeftID>
         <SiblingLeftName>"Reisende spezifizieren"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>"__TSK_0002_01_0006"<SiblingRightID>
         <SiblingRightName>"Reiseverbindungen anfordern"</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>Disabling</TemporalOperator>
       <UIConcepts></UIConcepts>
      </Subtask>
      <Subtask>
       <TaskID>"__TSK_0002_01_0006"</TaskID>
       <TaskName>"Reiseverbindungen anfordern"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Interaction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0002"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reisedaten spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>"__TSK_0002_01_0001"</ParentID>
         <ParentName>"Reisedaten spezifizieren"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0002_01_0005"</SiblingLeftID>
         <SiblingLeftName>"Reiseoptionen spezifizieren"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator></TemporalOperator>
       <UIConcepts>
        <UIConcept>
         <UIConceptID>"__CPT_0002_01_0001"</UIConceptID>
         <UIConceptName>"Reiseverbindungen anfordern"</UIConceptName>
        </UIConcept>
       </UIConcepts>
      </Subtask>
     </Subtasks>
    </Subtask>
   </TMF_Content>
  </MDFR_Fragment>
 </ModelFragment>
 <ModelFragment>
  <MDFR_Type>Concept</MDFR_Type>
  <MDFR_FragmentID>"__CMF_0002_01"</MDFR_FragmentID>
  <MDFR_Label>"Konzepte - Reisedaten spezifizieren"</MDFR_Label>
  <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
  </MDFR_Purpose>
  <MDFR_Annotation>""</MDFR_Annotation>
  <MDFR_Diagram>
```

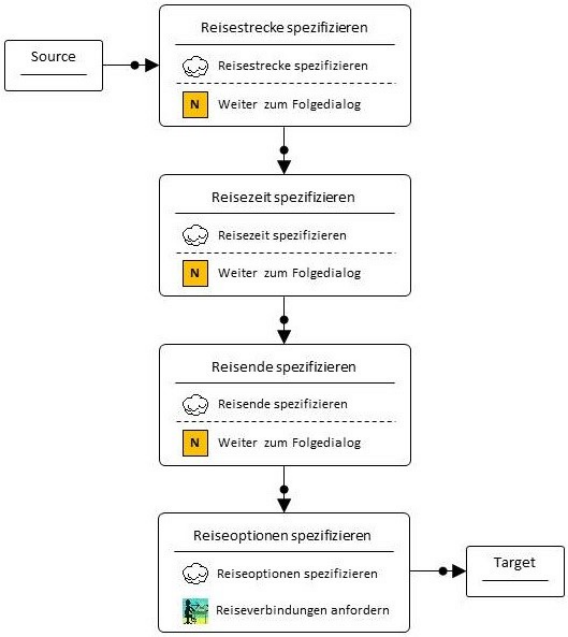
Konzepte des Patterns „Reisedaten spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0002_01_0001	Reiseverbindungen anfordern	Activator
2	__CPT_0002_01_0002	Weiter zum Folgedialog (1)	Navigator
3	__CPT_0002_01_0003	Weiter zum Folgedialog (2)	Navigator
4	__CPT_0002_01_0004	Weiter zum Folgedialog (3)	Navigator

```
</MDFR_Diagram>
<MDFR_Fragment>
 <CMF_IncludesDummy>False</CMF_IncludesDummy>
 <CMF_Content>
  <Concept>
   <CCPT_ConceptID>"__CPT_0002_01_0001"</CCPT_ConceptID>
   <CCPT_ConceptName>"Reiseverbindungen anfordern"</CCPT_ConceptName>
   <CCPT_Description>"Senden der Reisedaten und Anfordern der entsprechenden
                      Reiseverbindungen"</CCPT_Description>
   <CCPT_Label>"Daten senden"</CCPT_Label>
   <CCPT_Perceptible>True</CCPT_Perceptible>
   <CCPT_Enabled>True</CCPT_Enabled>
   <CCPT_Required>True</CCPT_Required>
   <CCPT_ConceptType>Activator</CCPT_ConceptType>
   <CCPT_DataType>""</CCPT_DataType>
   <CCPT_Value>""</CCPT_Value>
   <CCPT_ConceptOrigin>
    <CCPT_OriginPatternID>"PPT_0001_0002"</CCPT_OriginPatternID>
    <CCPT_OriginPatternName>"Reisedaten spezifizieren"</CCPT_OriginPatternName>
    <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
    <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
   </CCPT_ConceptOrigin>
   <CCPT_Preconditions></CCPT_Preconditions>
   <CCPT_Postconditions><CCPT_Postconditions>
   <CCPT_DataLink></CCPT_DataLink>
   <CCPT_TaskLinks>
    <CCPT_TaskLink>
     <CCPT_TaskID>"__TSK_0002_01_0006"</CCPT_TaskID>
     <CCPT_TaskName>"Reiseverbindungen anfordern"</CCPT_TaskName>
    </CCPT_TaskLink>
   </CCPT_TaskLinks>
  </Concept>
  <Concept>
   <CCPT_ConceptID>"__CPT_0002_01_0002"</CCPT_ConceptID>
   <CCPT_ConceptName>"Weiter zum Folgedialog (1)"</CCPT_ConceptName>
   <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                      modellierung im Dialog-Modell-Fragment benötigt wird"
   </CCPT_Description>
   <CCPT_Label>"Weiter"</CCPT_Label>
   <CCPT_Perceptible>True</CCPT_Perceptible>
   <CCPT_Enabled>True</CCPT_Enabled>
   <CCPT_Required>True</CCPT_Required>
   <CCPT_ConceptType>Navigator</CCPT_ConceptType>
   <CCPT_DataType>""</CCPT_DataType>
   <CCPT_Value>""</CCPT_Value>
   <CCPT_ConceptOrigin>
    <CCPT_OriginPatternID>"PPT_0001_0002"</CCPT_OriginPatternID>
    <CCPT_OriginPatternName>"Reisedaten spezifizieren"</CCPT_OriginPatternName>
    <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
    <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
   </CCPT_ConceptOrigin>
   <CCPT_Preconditions></CCPT_Preconditions>
   <CCPT_Postconditions><CCPT_Postconditions>
   <CCPT_DataLink></CCPT_DataLink>
   <CCPT_TaskLinks></CCPT_TaskLinks>
  </Concept>
  <Concept>
   <CCPT_ConceptID>"__CPT_0002_01_0003"</CCPT_ConceptID>
   <CCPT_ConceptName>"Weiter zum Folgedialog (2)"</CCPT_ConceptName>
   <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                      modellierung im Dialog-Modell-Fragment benötigt wird"
   </CCPT_Description>
   <CCPT_Label>"Weiter"</CCPT_Label>
   <CCPT_Perceptible>True</CCPT_Perceptible>
   <CCPT_Enabled>True</CCPT_Enabled>
   <CCPT_Required>True</CCPT_Required>
```

```
    <CCPT_ConceptType>Navigator</CCPT_ConceptType>
    <CCPT_DataType>""</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0002"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisedaten spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks></CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0002_01_0004"</CCPT_ConceptID>
    <CCPT_ConceptName>"Weiter zum Folgedialog (3)"</CCPT_ConceptName>
    <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                       modellierung im Dialog-Modell-Fragment benötigt wird"
    </CCPT_Description>
    <CCPT_Label>"Weiter"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>Navigator</CCPT_ConceptType>
    <CCPT_DataType>""</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0002"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisedaten spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks></CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0002_01"</MDFR_FragmentID>
 <MDFR_Label>"Reisedaten spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifizieren der Reisedaten"</MDFR_Purpose>
 <MDFR_Annotation>"Für den Kontext - Großer Bildschirm -"</MDFR_Annotation>
 <MDFR_Diagram>
```


```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0002_01_0001"</DialogID>
```

```
    <DialogName>"Reisedaten spezifizieren"</DialogName>
    <DialogDescription>""</DialogDescription>
    <DialogLabel>"Reisedaten spezifizieren"</DialogLabel>
    <DialogType>Complex</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED_DialogID>""</PRED_DialogID>
       <PRED_DialogName>""</PRED_DialogName>
      </Predecessor>
     </Predecessors>
     <Successors>
      <Successor>
       <SUCC_DialogID>""</SUCC_DialogID>
       <SUCC_DialogName>""</SUCC_DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>"__CPT_0002_01_0001"</SUCC_ConceptID>
        <SUCC_ConceptName>"Reiseverbindungen anfordern"</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0002_01_0002"</DLG_TaskID>
      <DLG_TaskName>"Reisestrecke spezifizieren"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     </DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0002_01_0003"</DLG_TaskID>
      <DLG_TaskName>"Reisezeit spezifizieren"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     </DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0002_01_0004"</DLG_TaskID>
      <DLG_TaskName>"Reisende spezifizieren"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     </DLG_Task>
    </DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0002_01_0005"</DLG_TaskID>
      <DLG_TaskName>"Reiseoptionen spezifizieren"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     </DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0002_01_0006"</DLG_TaskID>
      <DLG_TaskName>"Reiseverbindungen anfordern"</DLG_TaskName>
      <DLG_Processing>Exclusive</DLG_Processing>
     </DLG_Task>
    </DLG_Tasks>
   </Dialog>
  </DMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0002_02"</MDFR_FragmentID>
 <MDFR_Label>"Reisedaten spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifizieren der Reisedaten"</MDFR_Purpose>
 <MDFR_Annotation>"Für den Kontext - Kleiner Bildschirm -"</MDFR_Annotation>
 <MDFR_Diagram>
```




```
</MDFR_Diagram>
<MDFR_Fragment>
 <DMF_ContextModelReferences>
  <DMF_ContextModelReference>
   <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
   <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
   <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
   <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
  </DMF_ContextModelReference>
 </DMF_ContextModelReferences>
 <DMF_Content>
  <Dialog>
   <DialogID>"__DLG_0002_02_0001"</DialogID>
   <DialogName>"Reisestrecke spezifizieren"</DialogName>
   <DialogDescription>""</DialogDescription>
   <DialogLabel>"Reisestrecke spezifizieren"</DialogLabel>
   <DialogType>Complex</DialogType>
   <Position>
    <Predecessors>
     <Predecessor>
      <PRED_DialogID>""</PRED_DialogID>
      <PRED_DialogName>""</PRED_DialogName>
     </Predecessor>
    </Predecessors>
    <Successors>
     <Successor>
      <SUCC_DialogID>"__DLG_0002_02_0002"</SUCC_DialogID>
      <SUCC_DialogName>"Reisezeit spezifizieren"</SUCC_DialogName>
      <SUCC_TransitionType>Sequential</SUCC_TransitionType>
      <SUCC_Trigger>
       <SUCC_ConceptID>"__CPT_0002_02_0002"</SUCC_ConceptID>
       <SUCC_ConceptName>"Weiter zum Folgedialog (1)"</SUCC_ConceptName>
      </SUCC_Trigger>
     </Successor>
    </Successors>
   </Position>
   <DLG_Tasks>
    <DLG_Task>
     <DLG_TaskID>"__TSK_0002_01_0002"</DLG_TaskID>
     <DLG_TaskName>"Reisestrecke spezifizieren"</DLG_TaskName>
     <DLG_Processing>Recursive</DLG_Processing>
    </DLG_Task>
   </DLG_Tasks>
   <SupplementaryConcepts>
    <SupplementaryConcept>
```

```
   <SupplementaryConceptID>"__CPT_0002_01_0002"</SupplementaryConceptID>
   <SupplementaryConceptName>"Weiter zum Folgedialog (1)"
   </SupplementaryConceptName>
  </SupplementaryConcept>
 </SupplementaryConcepts>
</Dialog>
<Dialog>
 <DialogID>"__DLG_0002_02_0002"</DialogID>
 <DialogName>"Reisezeit spezifizieren"</DialogName>
 <DialogDescription>""</DialogDescription>
 <DialogLabel>"Reisezeit spezifizieren"</DialogLabel>
 <DialogType>Complex</DialogType>
 <Position>
  <Predecessors>
   <Predecessor>
    <PRED_DialogID>"__DLG_0002_02_0001"</PRED_DialogID>
    <PRED_DialogName>"Reisestrecke spezifizieren"</PRED_DialogName>
   </Predecessor>
  </Predecessors>
  <Successors>
   <Successor>
    <SUCC_DialogID>"__DLG_0002_02_0003"</SUCC_DialogID>
    <SUCC_DialogName>"Reisende spezifizieren"</SUCC_DialogName>
    <SUCC_TransitionType>Sequential</SUCC_TransitionType>
    <SUCC_Trigger>
     <SUCC_ConceptID>"__CPT_0002_01_0003"</SUCC_ConceptID>
     <SUCC_ConceptName>"Weiter zum Folgedialog (2)"</SUCC_ConceptName>
    </SUCC_Trigger>
   </Successor>
  </Successors>
 </Position>
 <DLG_Tasks>
  <DLG_Task>
   <DLG_TaskID>"__TSK_0002_01_0003"</DLG_TaskID>
   <DLG_TaskName>"Reisezeit spezifizieren"</DLG_TaskName>
   <DLG_Processing>Recursive</DLG_Processing>
  </DLG_Task>
 </DLG_Tasks>
 <SupplementaryConcepts>
  <SupplementaryConcept>
   <SupplementaryConceptID>"__CPT_0002_01_0003"</SupplementaryConceptID>
   <SupplementaryConceptName>"Weiter zum Folgedialog (2)"
   </SupplementaryConceptName>
  </SupplementaryConcept>
 </SupplementaryConcepts>
</Dialog>
<Dialog>
 <DialogID>"__DLG_0002_02_0003"</DialogID>
 <DialogName>"Reisende spezifizieren"</DialogName>
 <DialogDescription>""</DialogDescription>
 <DialogLabel>"Reisende spezifizieren"</DialogLabel>
 <DialogType>Complex</DialogType>
 <Position>
  <Predecessors>
   <Predecessor>
    <PRED_DialogID>"__DLG_0002_02_0002"</PRED_DialogID>
    <PRED_DialogName>"Reisezeit spezifizieren"</PRED_DialogName>
   </Predecessor>
  </Predecessors>
  <Successors>
   <Successor>
    <SUCC_DialogID>"__DLG_0002_02_0004"</SUCC_DialogID>
    <SUCC_DialogName>"Reiseoptionen spezifizieren"</SUCC_DialogName>
    <SUCC_TransitionType>Sequential</SUCC_TransitionType>
    <SUCC_Trigger>
     <SUCC_ConceptID>"__CPT_0002_01_0004"</SUCC_ConceptID>
     <SUCC_ConceptName>"Weiter zum Folgedialog (3)"</SUCC_ConceptName>
    </SUCC_Trigger>
   </Successor>
  </Successors>
 </Position>
```

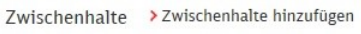
```
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0002_01_0004"</DLG_TaskID>
          <DLG_TaskName>"Reisende spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0002_01_0004"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog (3)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
       <Dialog>
        <DialogID>"__DLG_0002_02_0004"</DialogID>
        <DialogName>"Reiseoptionen spezifizieren"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Reiseoptionen spezifizieren"</DialogLabel>
        <DialogType>Complex</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>"__DLG_0002_02_0003"</PRED_DialogID>
           <PRED_DialogName>"Reisende spezifizieren"</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>"__CPT_0002_01_0001"</SUCC_ConceptID>
            <SUCC_ConceptName>"Reiseverbindungen anfordern"</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0002_01_0005"</DLG_TaskID>
          <DLG_TaskName>"Reiseoptionen spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0002_01_0006"</DLG_TaskID>
          <DLG_TaskName>"Reiseverbindungen anfordern"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.3 Reisestrecke spezifizieren

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0003"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reisestrecke spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0003_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind diejenigen Daten zu spezifizieren, die die gewünschte Reise-
                  strecke festlegen."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zur Reisestrecke werden benötigt:
                  1. Angabe des Startorts (von)
                  2. Angabe des Zielorts (nach)
                  3. Optionale Angabe eines Zwischenstopps (über)
                  4. Angabe der Reiserichtung, d. h. ob nur die Hinreise oder auch
                     eine Rückreise gewünscht wird"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
```

```
 <KnownUses>
  <KnownUse>
   <KNWN_KnownUseID>"0003_0001"</KNWN_KnownUseID>
   <KNWN_Label>"Angaben zur Reisestrecke - Deutsche Bahn AG"</KNWN_Label>
   <KNWN_Description>"Angabe der Reisestrecke im Online-Portal der Deutschen Bahn
                      (www.bahn.de)"</KNWN_Description>
   <KNWN_Images>
    <KNWN_Image>
```


```
    </KNWN_Image>
    <KNWN_Image>
```



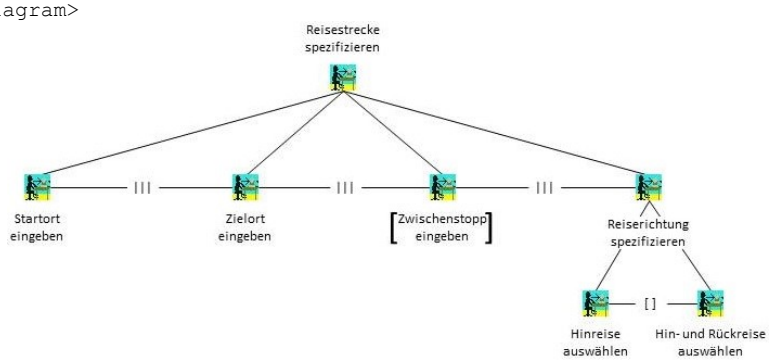

```
    </KNWN_Image>
    <KNWN_Image>
```


```
    </KNWN_Image>
   </KNWN_Images>
  </KnownUse>
  <KnownUse>
   <KNWN_KnownUseID>"0003_0002"</KNWN_KnownUseID>
   <KNWN_Label>"Angaben zur Reisestrecke - Deutsche Lufthansa AG"</KNWN_Label>
   <KNWN_Description>"Angabe der Reisestrecke im Online-Portal der Lufthansa
                      (www.lufthansa.de)"</KNWN_Description>
   <KNWN_Images>
    <KNWN_Image>
```


```
    </KNWN_Image>
   </KNWN_Images>
  </KnownUse>
  <KnownUse>
   <KNWN_KnownUseID>"0003_0003"</KNWN_KnownUseID>
   <KNWN_Label>"Angaben zur Reisestrecke - DB-Fahrkartenautomat"</KNWN_Label>
   <KNWN_Description>"Angabe der Reisestrecke an einem Fahrkartenautomat der
                      Deutschen Bahn"</KNWN_Description>
   <KNWN_Images>
    <KNWN_Image>
```


```
    </KNWN_Image>
   </KNWN_Images>
  </KnownUse>
 </KnownUses>
</Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```


```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0003_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0002"</RLTN_PatternID>
    <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0003_01"</MDFR_FragmentID>
    <MDFR_Label>"Reisestrecke spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren aller relevanten Parameter der Reise-
                   strecke"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>False</IncludesDummy>
     <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0003_01_0001"</TaskID>
       <TaskName>"Reisestrecke spezifizieren"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Interaction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reisestrecke spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
```

```
 <Parent>
  <ParentID>""</ParentID>
  <ParentName>""</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>""</SiblingLeftID>
  <SiblingLeftName>""</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>""<SiblingRightID>
  <SiblingRightName>""</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>Interleaving</TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0003_01_0002"</TaskID>
  <TaskName>"Startort eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Interaction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisestrecke spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0003_01_0001"</ParentID>
    <ParentName>"Reisestrecke spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0003_01_0003"<SiblingRightID>
    <SiblingRightName>"Zielort eingeben"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0003_01_0001"</UIConceptID>
    <UIConceptName>"Startort"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0003_01_0003"</TaskID>
  <TaskName>"Zielort eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Interaction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisestrecke spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
```

```
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0003_01_0001"</ParentID>
   <ParentName>"Reisestrecke spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0003_01_0002"</SiblingLeftID>
   <SiblingLeftName>"Startort eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0003_01_0004"<SiblingRightID>
   <SiblingRightName>"Zwischenstopp eingeben"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0003_01_0002"</UIConceptID>
   <UIConceptName>"Zielort"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0003_01_0004"</TaskID>
 <TaskName>"Zwischenstopp eingeben"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisestrecke spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>True</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0003_01_0001"</ParentID>
   <ParentName>"Reisestrecke spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0003_01_0003"</SiblingLeftID>
   <SiblingLeftName>"Zielort eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0003_01_0005"<SiblingRightID>
   <SiblingRightName>"Reiserichtung spezifizieren"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0003_01_0003"</UIConceptID>
   <UIConceptName>"Zwischenstopp"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0003_01_0005"</TaskID>
 <TaskName>"Reiserichtung spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisestrecke spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
```

```
 <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
</TaskOrigin>
<ContextsOfUse></ContextsOfUse>
<Optional>False</Optional>
<Iterative>False</Iterative>
<Conditional>False</Conditional>
<Preconditions></Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
  <ParentID>"__TSK_0003_01_0001"</ParentID>
  <ParentName>"Reisestrecke spezifizieren"</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>"__TSK_0003_01_0004"</SiblingLeftID>
  <SiblingLeftName>"Zwischenstopp eingeben"</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>""<SiblingRightID>
  <SiblingRightName>""</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator></TemporalOperator>
<UIConcepts>
 <UIConcept>
  <UIConceptID>"__CPT_0003_01_0004"</UIConceptID>
  <UIConceptName>"Reiserichtung"</UIConceptName>
 </UIConcept
</UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0003_01_0006"</TaskID>
  <TaskName>"Hinreise auswählen"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Interaction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisestrecke spezifizieren"
   </TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>True</Conditional>
  <Preconditions></Preconditions>
  <Postconditions>
   <Postcondition>
    <Operator>"assignment"</Operator>
    <Operands>
     <Operand>
      <OperandType>"Concept"</OperandType>
      <OperandID>"__CPT_0003_01_0004"</OperandID>
      <OperandName>"Reiserichtung"</OperandName>
      <OperandElement>"CCPT_Value"</OperandElement>
     </Operand>
     <Operand>
      <OperandType>"Constant"</OperandType>
      <OperandValue>"Hinreise"</OperandValue>
     </Operand>
    </Operands>
   </Postcondition>
  </Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0003_01_0005"</ParentID>
    <ParentName>"Reiserichtung spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
```

```
   <SiblingLeftName>""</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0003_01_0005"<SiblingRightID>
   <SiblingRightName>"Hin- und Rückreise wählen"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Choice</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0003_01_0005"</UIConceptID>
   <UIConceptName>"Hinreise"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0003_01_0007"</TaskID>
 <TaskName>"Hin- und Rückreise auswählen"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0003"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisestrecke spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
  <Postcondition>
   <Operator>"assignment"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0003_01_0004"</OperandID>
     <OperandName>"Reiserichtung"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Hin- und Rückreise"</OperandValue>
    </Operand>
   </Operands>
  </Postcondition>
 </Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0003_01_0005"</ParentID>
   <ParentName>"Reiserichtung spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0003_01_0006"</SiblingLeftID>
   <SiblingLeftName>"Hinreise wählen"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>""<SiblingRightID>
   <SiblingRightName>""</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator></TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0003_01_0006"</UIConceptID>
   <UIConceptName>"Hin- und Rückreise"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
```

```
      </Subtasks>
     </Subtask>
    </Subtasks>
   </Subtask>
  </TMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0003_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Reisestrecke spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reisestrecke spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0003_01_0001	Startort	DataEdit
2	__CPT_0003_01_0002	Zielort	DataEdit
3	__CPT_0003_01_0003	Zwischenstopp	DataEdit
4	__CPT_0003_01_0004	Reiserichtung	SingleChoice
5	__CPT_0003_01_0005	Hinreise	ChoiceItem
6	__CPT_0003_01_0006	Hin- und Rückreise	ChoiceItem

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>False</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0003_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Startort"</CCPT_ConceptName>
    <CCPT_Description>"Eingabe des Startortes"</CCPT_Description>
    <CCPT_Label>"Startort:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisestrecke spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0003_01_0002"</CCPT_TaskID>
      <CCPT_TaskName>"Startort eingeben"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0003_01_0002"</CCPT_ConceptID>
    <CCPT_ConceptName>"Zielort"</CCPT_ConceptName>
    <CCPT_Description>"Eingabe des Zielortes"</CCPT_Description>
    <CCPT_Label>"Zielort:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisestrecke spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
```

```
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0003_01_0003"</CCPT_TaskID>
   <CCPT_TaskName>"Zielort eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0003_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Zwischenstopp"</CCPT_ConceptName>
 <CCPT_Description>"Eingabe des Zwischenstopps"</CCPT_Description>
 <CCPT_Label>"Zwischenstopp:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisestrecke spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0003_01_0004"</CCPT_TaskID>
   <CCPT_TaskName>"Zwischenstopp eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0003_01_0004"</CCPT_ConceptID>
 <CCPT_ConceptName>"Reiserichtung"</CCPT_ConceptName>
 <CCPT_Description>"Auswahl der Reiserichtung"</CCPT_Description>
 <CCPT_Label>"Reiserichtung:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"Integer"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisestrecke spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0003_01_0005"</CCPT_TaskID>
   <CCPT_TaskName>"Reiserichtung spezifizieren"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0003_01_0005"</CCPT_ConceptID>
 <CCPT_ConceptName>"Hinreise"</CCPT_ConceptName>
 <CCPT_Description>"Nur Hinreise auswählen"</CCPT_Description>
 <CCPT_Label>"Hinreise"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
```

```
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisestrecke spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0003_01_0005"</CCPT_TaskID>
      <CCPT_TaskName>"Hinreise auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0003_01_0006"</CCPT_ConceptID>
    <CCPT_ConceptName>"Hin- und Rückreise"</CCPT_ConceptName>
    <CCPT_Description>"Hin- und Rückreise auswählen"</CCPT_Description>
    <CCPT_Label>"Hin- und Rückreise"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0003"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisestrecke spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0003_01_0007"</CCPT_TaskID>
      <CCPT_TaskName>"Hin- und Rückreise auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0003_01"</MDFR_FragmentID>
 <MDFR_Label>"Reisestrecke spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Angabe der Orte und Reiserichtung"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```


```
 </MDFR_Diagram>
 <MDFR_Fragment>
```

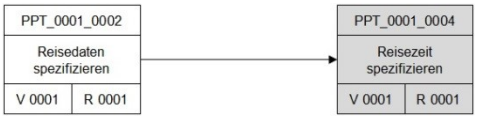
```
      <DMF_IncludesDummy>False</DMF_IncludesDummy>
      <DMF_ContextModelReferences>
       <DMF_ContextModelReference>
        <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
        <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
        <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
        <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
       </DMF_ContextModelReference>
       <DMF_ContextModelReference>
        <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
        <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
        <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
        <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
       </DMF_ContextModelReference>
      </DMF_ContextModelReferences>
      <DMF_Content>
       <Dialog>
        <DialogID>"__DLG_0003_01_0001"</DialogID>
        <DialogName>"Reisestrecke spezifizieren"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Reisestrecke"</DialogLabel>
        <DialogType>Single</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED DialogID>""</PRED DialogID>
           <PRED_DialogName>""</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC DialogID>""</SUCC DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC ConceptTaskID>""</SUCC ConceptID>
            <SUCC_ConceptName>""</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0003_01_0002"</DLG_TaskID>
          <DLG_TaskName>"Startort eingeben"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0003_01_0003"</DLG_TaskID>
          <DLG_TaskName>"Zielort eingeben"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0003_01_0004"</DLG_TaskID>
          <DLG_TaskName>"Zwischenstopp eingeben"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0003_01_0005"</DLG_TaskID>
          <DLG_TaskName>"Reiserichtung spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.4 Reisezeit spezifizieren

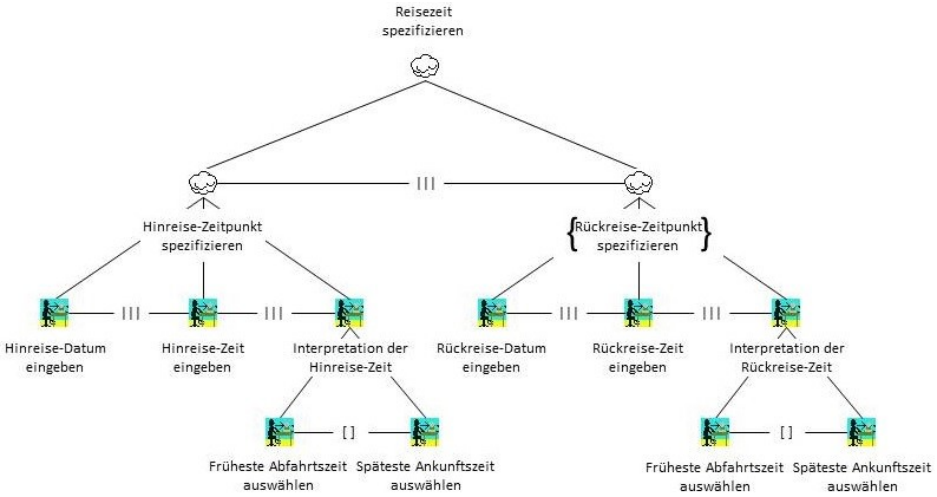
```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0004"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reisezeit spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0004_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind diejenigen Daten zu spezifizieren, die die gewünschte
                  Reisezeit festlegen."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zur Reisezeit werden benötigt:
                  1. Angabe des Hinreisezeitpunkts
                  1.1. Datum der Hinreise
                  1.2. Uhrzeit der Hinreise
                  1.3. Angabe, ob der angegebene Hinreise-Zeitpunkt als Abfahrtszeit
                       oder Ankunftszeit zu interpretieren ist
                  2. Optionale Angabe des Rückreisezeitpunkts
                  2.1. Datum der Rückreise
                  2.2. Uhrzeit der Rückreise
```

```
                2.3. Angabe, ob der angegebene Rückreise-Zeitpunkt als Abfahrtszeit
                     oder Ankunftszeit zu interpretieren ist"</SLTN_Digest>
  </Solution>
 </Theory>
 <Practice>
  <KnownUses>
   <KnownUse>
    <KNWN_KnownUseID>"0004_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Angaben zur Reisestrecke - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Angabe der Reisezeit im Online-Portal der Deutschen Bahn
                       (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```


```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0004_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0002"</RLTN_PatternID>
    <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0004_01"</MDFR_FragmentID>
    <MDFR_Label>"Reisezeit spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren aller relevanten Parameter der Reisezeit"
    </MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>False</IncludesDummy>
 <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0004_01_0001"</TaskID>
   <TaskName>"Reisezeit spezifizieren"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
    <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Interleaving</TemporalOperator>
   <UIConcepts></UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0004_01_0002"</TaskID>
     <TaskName>"Hinreise-Zeitpunkt spezifizieren"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Abstraction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
      <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
```

```
<Optional>False</Optional>
<Iterative>False</Iterative>
<Conditional>False</Conditional>
<Preconditions></Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
  <ParentID>"__TSK_0004_01_0001"</ParentID>
  <ParentName>"Reisezeit spezifizieren"</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>""</SiblingLeftID>
  <SiblingLeftName>""</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>"__TSK_0004_01_0003"<SiblingRightID>
  <SiblingRightName>"Rückreisezeitpunkt spezifizieren"</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>Interleaving</TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0004_01_0004"</TaskID>
  <TaskName>"Hinreise-Datum eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Interaction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False"</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0004_01_0002"</ParentID>
    <ParentName>"Hinreise-Zeitpunkt spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0004_01_0005"<SiblingRightID>
    <SiblingRightName>"Hinreise-Zeit eingeben"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0004_01_0001"</UIConceptID>
    <UIConceptName>"Hinreisedatum"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0004_01_0005"</TaskID>
  <TaskName>"Hinreise-Zeit eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>"Interaction"</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
```

```
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0002"</ParentID>
   <ParentName>"Hinreise-Zeitpunkt spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0004_01_0004"</SiblingLeftID>
   <SiblingLeftName>"Hinreise-Datum eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0004_01_0006"<SiblingRightID>
   <SiblingRightName>"Interpretation der Hinreise-Zeit"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0002"</UIConceptID>
   <UIConceptName>"Hinreisezeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0004_01_0006"</TaskID>
 <TaskName>"Interpretation der Hinreise-Zeit"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
<TaskOrigin>
 <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
 <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
 <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
 <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
</TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0002"</ParentID>
   <ParentName>"Hinreise-Zeitpunkt spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0004_01_0005"</SiblingLeftID>
   <SiblingLeftName>"Hinreise-Zeit eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>""<SiblingRightID>
   <SiblingRightName>""</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator></TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0003"</UIConceptID>
   <UIConceptName>"Interpretation Hinreisezeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
 <Subtasks>
  <Subtask>
   <TaskID>"__TSK_0004_01_0007"</TaskID>
   <TaskName>"Früheste Abfahrtszeit auswählen (Hinreise)"</TaskName>
```

```
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
  <Postcondition>
   <Operator>"assignment"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0004_01_0003"</OperandID>
     <OperandName>"Interpretation Hinreisezeit"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Abfahrtszeit"</OperandValue>
    </Operand>
   </Operands>
  </Postcondition>
 </Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0006"</ParentID>
   <ParentName>"Interpretation der Hinreise-Zeit"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>""</SiblingLeftID>
   <SiblingLeftName>""</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0004_01_0008"<SiblingRightID>
   <SiblingRightName>"Späteste Ankunftszeit wählen"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Choice</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0007"</UIConceptID>
   <UIConceptName>"Hinreise Abfahrtszeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0004_01_0008"</TaskID>
 <TaskName>"Späteste Ankunftszeit auswählen (Hinreise)"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
```

```
      <Postcondition>
       <Operator>"assignment"</Operator>
       <Operands>
        <Operand>
         <OperandType>"Concept"</OperandType>
         <OperandID>"__CPT_0004_01_0003"</OperandID>
         <OperandName>"Interpretation Hinreisezeit"</OperandName>
         <OperandElement>"CCPT_Value"</OperandElement>
        </Operand>
        <Operand>
         <OperandType>"Constant"</OperandType>
         <OperandValue>"Ankunftszeit"</OperandValue>
        </Operand>
       </Operands>
      </Postcondition>
     </Postconditions>
     <Position>
      <Parent>
       <ParentID>"__TSK_0004_01_0006"</ParentID>
       <ParentName>"Interpretation der Hinreise-Zeit"</ParentName>
      </Parent>
      <SiblingLeft>
       <SiblingLeftID>"__TSK_0004_01_0007"</SiblingLeftID>
       <SiblingLeftName>"Früheste Abfahrtszeit auswählen (Hinreise)"
       </SiblingLeftName>
      </SiblingLeft>
      <SiblingRight>
       <SiblingRightID>""<SiblingRightID>
       <SiblingRightName>""</SiblingRightName>
      </SiblingRight>
     </Position>
     <TemporalOperator></TemporalOperator>
     <UIConcepts>
      <UIConcept>
       <UIConceptID>"__CPT_0004_01_0008"</UIConceptID>
       <UIConceptName>"Hinreise Ankunftszeit"</UIConceptName>
      </UIConcept
     </UIConcepts>
    </Subtask>
   <Subtasks>
  </Subtask>
 </Subtasks>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0004_01_0003"</TaskID>
 <TaskName>"Rückreise-Zeitpunkt spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions>
  <Precondition>
   <Operator>"equal"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0003_01_0004"</OperandID>
     <OperandName>"Reiserichtung"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Hin- und Rückreise"</OperandValue>
```

```
   </Operand>
  </Operands>
 </Precondition
</Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
  <ParentID>"__TSK_0004_01_0001"</ParentID>
  <ParentName>"Reisezeit spezifizieren"</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>"__TSK_0004_01_0002"</SiblingLeftID>
  <SiblingLeftName>"Hinreisezeitpunkt spezifizieren"</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>""<SiblingRightID>
  <SiblingRightName>""</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator></TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0004_01_0009"</TaskID>
  <TaskName>"Rückreise-Datum eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>"Interaction"</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>True</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0004_01_0003"</ParentID>
    <ParentName>"Rückreise-Zeitpunkt spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0004_01_0010"<SiblingRightID>
    <SiblingRightName>"Rückreise-Zeit eingeben"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving"/TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0004_01_0004"</UIConceptID>
    <UIConceptName>"Rückreisedatum"</UIConceptName>
   </UIConcept
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"__TSK_0004_01_0010"</TaskID>
  <TaskName>"Rückreise-Zeit eingeben"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Interaction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
```

```
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0003"</ParentID>
   <ParentName>"Rückreise-Zeitpunkt spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0004_01_0009"</SiblingLeftID>
   <SiblingLeftName>"Rückreise-Datum eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0004_01_0011"<SiblingRightID>
   <SiblingRightName>"Interpretation der Rückreise-Zeit"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0005"</UIConceptID>
   <UIConceptName>"Rückreisezeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0004_01_0011"</TaskID>
 <TaskName>"Interpretation der Rückreise-Zeit"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>"Interaction"</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0003"</ParentID>
   <ParentName>"Rückreise-Zeitpunkt spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0004_01_0010"</SiblingLeftID>
   <SiblingLeftName>"Rückreise-Zeit eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>""<SiblingRightID>
   <SiblingRightName>""</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator></TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0006"</UIConceptID>
   <UIConceptName>"Interpretation Rückreisezeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
 <Subtasks>
  <Subtask>
   <TaskID>"__TSK_0004_01_0012"</TaskID>
   <TaskName>"Früheste Abfahrtszeit auswählen (Rückreise)"</TaskName>
```

```
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
  <Postcondition>
   <Operator>"assignment"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0004_01_0006"</OperandID>
     <OperandName>"Interpretation Rückreisezeit"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Abfahrtszeit"</OperandValue>
    </Operand>
   </Operands>
  </Postcondition>
 </Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0004_01_0011"</ParentID>
   <ParentName>"Interpretation der Rückreise-Zeit"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>""</SiblingLeftID>
   <SiblingLeftName>""</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0004_01_0012"<SiblingRightID>
   <SiblingRightName>"Späteste Ankunftszeit wählen"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Choice</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0004_01_0009"</UIConceptID>
   <UIConceptName>"Rückreise Abfahrtszeit"</UIConceptName>
  </UIConcept
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0004_01_0013"</TaskID>
 <TaskName>"Späteste Ankunftszeit auswählen (Rückreise)"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0004"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisezeit spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
```

```
          <Postcondition>
           <Operator>"assignment"</Operator>
           <Operands>
            <Operand>
             <OperandType>"Concept"</OperandType>
             <OperandID>"__CPT_0004_01_0003"</OperandID>
             <OperandName>"Interpretation Hinreisezeit"</OperandName>
             <OperandElement>"CCPT_Value"</OperandElement>
            </Operand>
            <Operand>
             <OperandType>"Constant"</OperandType>
             <OperandValue>"Ankunftszeit"</OperandValue>
            </Operand>
           </Operands>
          </Postcondition>
         </Postconditions>
         <Position>
          <Parent>
           <ParentID>"__TSK_0004_01_0011"</ParentID>
           <ParentName>"Interpretation der Rückreise-Zeit"</ParentName>
          </Parent>
          <SiblingLeft>
           <SiblingLeftID>"__TSK_0004_01_0012"</SiblingLeftID>
           <SiblingLeftName>"Früheste Abfahrtszeit auswählen (Rückreise)"
           </SiblingLeftName>
          </SiblingLeft>
          <SiblingRight>
           <SiblingRightID>""<SiblingRightID>
           <SiblingRightName>""</SiblingRightName>
          </SiblingRight>
         </Position>
         <TemporalOperator></TemporalOperator>
         <UIConcepts>
          <UIConcept>
           <UIConceptID>"__CPT_0004_01_0010"</UIConceptID>
           <UIConceptName>"Rückreise Ankunftszeit"</UIConceptName>
          </UIConcept
         </UIConcepts>
        </Subtask>
       <Subtasks>
      </Subtask>
     </Subtasks>
    </Subtask>
   </Subtasks>
  </Subtask>
 </TMF_Content>
</MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0004_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Reisezeit spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reisezeit spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0004_01_0001	Hinreisedatum	DataEdit
2	__CPT_0004_01_0002	Hinreisezeit	DataEdit
3	__CPT_0004_01_0003	Interpretation Hinreisezeit	SingleChoice
4	__CPT_0004_01_0004	Rückreisedatum	DataEdit
5	__CPT_0004_01_0005	Rückreisezeit	DataEdit
6	__CPT_0004_01_0006	Interpretation Rückreisezeit	SingleChoice
7	__CPT_0004_01_0007	Hinreise Abfahrtszeit	ChoiceItem
8	__CPT_0004_01_0008	Hinreise Ankunftszeit	ChoiceItem
9	__CPT_0004_01_0009	Rückreise Abfahrtszeit	ChoiceItem
10	__CPT_0004_01_0010	Rückreise Ankunftszeit	ChoiceItem

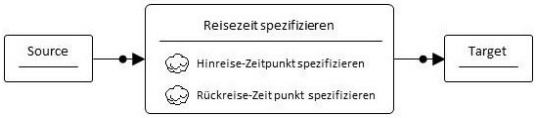
```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>False</CMF_IncludesDummy>
  <CMF_Content>
```

```
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0001"</CCPT_ConceptID>
 <CCPT_ConceptName>"Hinreisedatum"</CCPT_ConceptName>
 <CCPT_Description>"Eingabe des Datums der Hinreise"</CCPT_Description>
 <CCPT_Label>"Hinreise-Datum:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
 <CCPT_DataType>"Date"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0004"</CCPT_TaskID>
   <CCPT_TaskName>"Hinreise-Datum eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0002"</CCPT_ConceptID>
 <CCPT_ConceptName>"Hinreisezeit"</CCPT_ConceptName>
 <CCPT_Description>"Eingabe des Zeitpunkts der Hinreise"</CCPT_Description>
 <CCPT_Label>"Hinreise-Zeit:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
 <CCPT_DataType>"Time"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0005"</CCPT_TaskID>
   <CCPT_TaskName>"Hinreise-Zeit eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Interpretation Hinreisezeit"</CCPT_ConceptName>
 <CCPT_Description>"Abfahrts- oder Ankunftszeit"</CCPT_Description>
 <CCPT_Label>"Abfahrt / Ankunft:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
```

```
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0006"</CCPT_TaskID>
   <CCPT_TaskName>"Interpretation der Hinreise-Zeit"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0004"</CCPT_ConceptID>
 <CCPT_ConceptName>"Rückreisedatum"</CCPT_ConceptName>
 <CCPT_Description>"Eingabe des Datums der Rückreise"</CCPT_Description>
 <CCPT_Label>"Rückreise-Datum:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
     <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
 <CCPT_DataType>"Date"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0009"</CCPT_TaskID>
   <CCPT_TaskName>"Rückreise-Datum eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0005"</CCPT_ConceptID>
 <CCPT_ConceptName>"Rückreisezeit"</CCPT_ConceptName>
 <CCPT_Description>"Eingabe des Zeitpunkts der Rückreise"</CCPT_Description>
 <CCPT_Label>"Rückreise-Zeit:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
 <CCPT_DataType>"Time"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0010"</CCPT_TaskID>
   <CCPT_TaskName>"Rückreise-Zeit eingeben"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0006"</CCPT_ConceptID>
 <CCPT_ConceptName>"Interpretation Rückreisezeit"</CCPT_ConceptName>
 <CCPT_Description>"Ankunfts- oder Abfahrtszeit"</CCPT_Description>
 <CCPT_Label>"Abfahrt / Ankunft:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
```

```
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0011"</CCPT_TaskID>
   <CCPT_TaskName>"Interpretation der Rückreise-Zeit"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0007"</CCPT_ConceptID>
 <CCPT_ConceptName>"Hinreise Abfahrtszeit"</CCPT_ConceptName>
 <CCPT_Description>"Früheste Abfahrtszeit bei der Hinreise"</CCPT_Description>
 <CCPT_Label>"Früheste Abfahrtszeit"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0004_01_0007"</CCPT_TaskID>
   <CCPT_TaskName>"Früheste Abfahrtszeit auswählen (Hinreise)"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0004_01_0008"</CCPT_ConceptID>
 <CCPT_ConceptName>"Hinreise Ankunftszeit"</CCPT_ConceptName>
 <CCPT_Description>"Späteste Ankunftszeit bei der Hinreise"</CCPT_Description>
 <CCPT_Label>"Späteste Ankunftszeit"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reisezeit spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
```

```
      <CCPT_TaskID>"__TSK_0004_01_0008"</CCPT_TaskID>
      <CCPT_TaskName>"Späteste Ankunftszeit auswählen (Hinreise)"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0004_01_0009"</CCPT_ConceptID>
    <CCPT_ConceptName>"Rückreise Abfahrtszeit"</CCPT_ConceptName>
    <CCPT_Description>"Früheste Abfahrtszeit bei der Rückreise"</CCPT_Description>
    <CCPT_Label>"Früheste Abfahrtszeit"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisezeit spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0004_01_0012"</CCPT_TaskID>
      <CCPT_TaskName>"Früheste Abfahrtszeit auswählen (Rückreise)"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0004_01_0010"</CCPT_ConceptID>
    <CCPT_ConceptName>"Rückreise Ankunftszeit"</CCPT_ConceptName>
    <CCPT_Description>"Späteste Ankunftszeit bei der Rückreise"</CCPT_Description>
    <CCPT_Label>"Späteste Ankunftszeit"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0004"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisezeit spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0004_01_0013"</CCPT_TaskID>
      <CCPT_TaskName>"Späteste Ankunftszeit auswählen (Rückreise)"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0004_01"</MDFR_FragmentID>
 <MDFR_Label>"Reisezeit spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Angabe von Datum und Zeit der Reise"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```

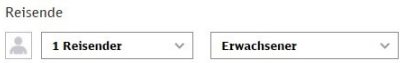



```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <DMF_IncludesDummy>False</DMF_IncludesDummy>
     <DMF_ContextModelReferences>
      <DMF_ContextModelReference>
       <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
       <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
       <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
       <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
      </DMF_ContextModelReference>
      <DMF_ContextModelReference>
       <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
       <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
       <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
       <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
      </DMF_ContextModelReference>
     </DMF_ContextModelReferences>
     <DMF_Content>
      <Dialog>
       <DialogID>"__DLG_0004_01_0001"</DialogID>
       <DialogName>"Reisezeit spezifizieren"</DialogName>
       <DialogDescription>""</DialogDescription>
       <DialogLabel>"Reisezeit spezifizieren"</DialogLabel>
       <DialogType>Single</DialogType>
       <Position>
        <Predecessors>
         <Predecessor>
          <PRED_DialogID>""</PRED_DialogID>
          <PRED_DialogName>""</PRED_DialogName>
         </Predecessor>
        </Predecessors>
        <Successors>
         <Successor>
          <SUCC_DialogID>""</SUCC_DialogID>
          <SUCC_DialogName>""</SUCC_DialogName>
          <SUCC_TransitionType>Sequential</SUCC_TransitionType>
          <SUCC_Trigger>
           <SUCC_ConceptID>""</SUCC_ConceptID>
           <SUCC_ConceptName>""</SUCC_ConceptName>
          </SUCC_Trigger>
         </Successor>
        </Successors>
       </Position>
       <DLG_Tasks>
        <DLG_Task>
         <DLG_TaskID>"__TSK_0004_01_0002"</DLG_TaskID>
         <DLG_TaskName>"Hinreise-Zeitpunkt spezifizieren"</DLG_TaskName>
         <DLG_Processing>Recursive</DLG_Processing>
        </DLG_Task>
        <DLG_Task>
         <DLG_TaskID>"__TSK_0004_01_0010"</DLG_TaskID>
         <DLG_TaskName>"Rückreise-Zeitpunkt spezifizieren"</DLG_TaskName>
         <DLG_Processing>Recursive</DLG_Processing>
        </DLG_Task>
       </DLG_Tasks>
      </Dialog>
     </DMF_Content>
    </MDFR_Fragment>
   </ModelFragment>
  </ModelFragments>
 </PaMGIS>
</Deployment>
</Pattern>
```

J.5 Reisende spezifizieren

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0005"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reisende spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0005_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind die tariflich relevanten Eigenschaften der zu befördernden
                  Personen und gegebenenfalls mitzunehmenden Dinge festzulegen."
    </PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zu den Reisenden werden benötigt:
                  1. Anzahl Erwachsene
                  2. Anzahl Kinder
                  3. Optionale Anzahlen hinsichtlich Personen aus weiteren Tarif-
                     gruppen oder tariflich relevanten Dingen, wie beispielsweise
                     Fahrräder oder Haustiere"</SLTN_Digest>
   </Solution>
  </Theory>
```

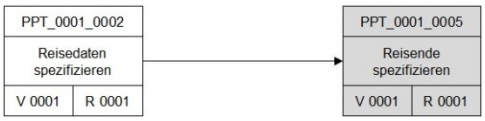
```
 <Practice>
  <KnownUses>
   <KnownUse>
    <KNWN_KnownUseID>"0005_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Angaben zu den Reisenden - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Angabe der Reisenden im Online-Portal der Deutschen Bahn
                       (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```




```
     </KNWN_Image>
     <KNWN_Image>
```

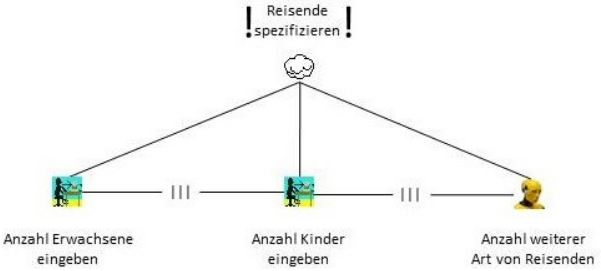

```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
   <KnownUse>
    <KNWN_KnownUseID>"0005_0002"</KNWN_KnownUseID>
    <KNWN_Label>"Angaben zu den Reisenden - Deutsche Lufthansa AG"</KNWN_Label>
    <KNWN_Description>"Angabe der Reisenden im Online-Portal der Lufthansa
                       (www.lufthansa.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```


```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0005_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0002"</RLTN_PatternID>
    <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
```

<MDFR_Type>Task</MDFR_Type>
<MDFR_FragmentID>"__TMF_0005_01"</MDFR_FragmentID>
<MDFR_Label>"Reisende spezifizieren"</MDFR_Label>
<MDFR_Purpose>"Spezifizieren aller relevanten Parameter bezüglich der Reisenden"
</MDFR_Purpose>
<MDFR_Annotation>""</MDFR_Annotation>
<MDFR_Diagram>



</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>True</IncludesDummy>
 <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0005_01_0001"</TaskID>
   <TaskName>"Reisende spezifizieren"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0005"</TaskOriginPatternID>
    <TaskOriginPatternName>"Reisende spezifizieren"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Interleaving</TemporalOperator>
   <UIConcepts></UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0005_01_0002"</TaskID>
     <TaskName>"Anzahl Erwachsene eingeben"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Interaction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0005"</TaskOriginPatternID>
      <TaskOriginPatternName>"Reisende spezifizieren"</TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>

```
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0005_01_0001"</ParentID>
   <ParentName>"Reisende spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>""</SiblingLeftID>
   <SiblingLeftName>""</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0005_01_0003"<SiblingRightID>
   <SiblingRightName>"Anzahl Kinder eingeben"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0005_01_0001"</UIConceptID>
   <UIConceptName>"Anzahl Erwachsene"</UIConceptName>
  </UIConcept>
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0005_01_0003"</TaskID>
 <TaskName>"Anzahl Kinder eingeben"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Interaction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0005"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reisende spezifizieren"</TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0005_01_0001"</ParentID>
   <ParentName>"Reisende spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0005_01_0002"</SiblingLeftID>
   <SiblingLeftName>"Anzahl Erwachsene eingeben"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"  TSK 0005 01 0004"<SiblingRightID>
   <SiblingRightName>"Anzahl weiterer Art von Reisenden eingeben"
   </SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Interleaving</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0005_01_0002"</UIConceptID>
   <UIConceptName>"Anzahl Kinder"</UIConceptName>
  </UIConcept>
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0005_01_0004"</TaskID>
 <TaskName>"Anzahl weiterer Art von Reisenden eingeben"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Dummy</TaskType>
 <TaskOrigin>
```

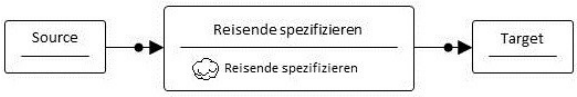
```
        <TaskOriginPatternID>"PPT_0001_0005"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reisende spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>"__TSK_0005_01_0001"</ParentID>
         <ParentName>"Reisende spezifizieren"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0005_01_0003"</SiblingLeftID>
         <SiblingLeftName>"Anzahl Kinder eingeben"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>Interleaving</TemporalOperator>
       <UIConcepts>
        <UIConcept>
         <UIConceptID>"__CPT_0005_01_0003"</UIConceptID>
         <UIConceptName>"Anzahl Dummy"</UIConceptName>
        </UIConcept>
       </UIConcepts>
      </Subtask>
     </Subtasks>
    </Subtask>
   </TMF_Content>
  </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0001_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Reisedaten spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reisende spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0005_01_0001	Anzahl Erwachsene	DataEdit
2	__CPT_0005_01_0002	Anzahl Kinder	DataEdit
3	__CPT_0005_01_0003	Anzahl Dummy	DataEdit

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>True</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0005_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Anzahl Erwachsene"</CCPT_ConceptName>
    <CCPT_Description>"Eingabe der Anzahl erwachsener Reisender"
    </CCPT_Description>
    <CCPT_Label>"Anzahl Erwachsene:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
    <CCPT_DataType>"Integer"</CCPT_DataType>
    <CCPT_Value>"1"</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0005"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reisende spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
```

```
      <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
     </CCPT_ConceptOrigin>
     <CCPT_Preconditions></CCPT_Preconditions>
     <CCPT_Postconditions><CCPT_Postconditions>
     <CCPT_DataLink></CCPT_DataLink>
     <CCPT_TaskLinks>
      <CCPT_TaskLink>
       <CCPT_TaskID>"__TSK_0005_01_0002"</CCPT_TaskID>
       <CCPT_TaskName>"Anzahl Erwachsene eingeben"</CCPT_TaskName>
      </CCPT_TaskLink>
     </CCPT_TaskLinks>
    </Concept>
    <Concept>
     <CCPT_ConceptID>"__CPT_0005_01_0002"</CCPT_ConceptID>
     <CCPT_ConceptName>"Anzahl Kinder"</CCPT_ConceptName>
     <CCPT_Description>"Eingabe der Anzahl reisender Kinder"</CCPT_Description>
     <CCPT_Label>"Anzahl Kinder:"</CCPT_Label>
     <CCPT_Perceptible>True</CCPT_Perceptible>
     <CCPT_Enabled>True</CCPT_Enabled>
     <CCPT_Required>True</CCPT_Required>
     <CCPT_ConceptType>DataEdit</CCPT_ConceptType>
     <CCPT_DataType>"Integer"</CCPT_DataType>
     <CCPT_Value>"0"</CCPT_Value>
     <CCPT_ConceptOrigin>
      <CCPT_OriginPatternID>"PPT_0001_0005"</CCPT_OriginPatternID>
      <CCPT_OriginPatternName>"Reisende spezifizieren"</CCPT_OriginPatternName>
      <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
      <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
     </CCPT_ConceptOrigin>
     <CCPT_Preconditions></CCPT_Preconditions>
     <CCPT_Postconditions><CCPT_Postconditions>
     <CCPT_DataLink></CCPT_DataLink>
     <CCPT_TaskLinks>
      <CCPT_TaskLink>
       <CCPT_TaskID>"__TSK_0005_01_0003"</CCPT_TaskID>
       <CCPT_TaskName>"Anzahl Kinder eingeben"</CCPT_TaskName>
      </CCPT_TaskLink>
     </CCPT_TaskLinks>
    </Concept>
    <Concept>
     <CCPT_ConceptID>"__CPT_0005_01_0003"</CCPT_ConceptID>
     <CCPT_ConceptName>"Anzahl Dummy"</CCPT_ConceptName>
     <CCPT_Description>"Eingabe der Anzahl weiterer Art von Reisenden"
     </CCPT_Description>
     <CCPT_Label>"Anzahl Dummy:"</CCPT_Label>
     <CCPT_Perceptible>True</CCPT_Perceptible>
     <CCPT_Enabled>True</CCPT_Enabled>
     <CCPT_Required>True</CCPT_Required>
     <CCPT_ConceptType>Dummy</CCPT_ConceptType>
     <CCPT_DataType>"Integer"</CCPT_DataType>
     <CCPT_Value>"0"</CCPT_Value>
     <CCPT_ConceptOrigin>
      <CCPT_OriginPatternID>"PPT_0001_0005"</CCPT_OriginPatternID>
      <CCPT_OriginPatternName>"Reisende spezifizieren"</CCPT_OriginPatternName>
      <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
      <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
     </CCPT_ConceptOrigin>
     <CCPT_Preconditions></CCPT_Preconditions>
     <CCPT_Postconditions><CCPT_Postconditions>
     <CCPT_DataLink></CCPT_DataLink>
     <CCPT_TaskLinks>
      <CCPT_TaskLink>
       <CCPT_TaskID>"__TSK_0005_01_0004"</CCPT_TaskID>
       <CCPT_TaskName>"Anzahl weiterer Art von Reisenden eingeben"</CCPT_TaskName>
      </CCPT_TaskLink>
     </CCPT_TaskLinks>
    </Concept>
   </CMF_Content>
  </MDFR_Fragment>
 </ModelFragment>
 <ModelFragment>
```

```
      <MDFR_Type>Dialog</MDFR_Type>
      <MDFR_FragmentID>"__DMF_0005_01"</MDFR_FragmentID>
      <MDFR_Label>"Reisende spezifizieren"</MDFR_Label>
      <MDFR_Purpose>"Angaben zu den Reisenden"</MDFR_Purpose>
      <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
      <MDFR_Diagram>
```




```
      </MDFR_Diagram>
      <MDFR_Fragment>
       <DMF_IncludesDummy>False</DMF_IncludesDummy>
       <DMF_ContextModelReferences>
        <DMF_ContextModelReference>
         <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
         <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
         <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
         <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
        </DMF_ContextModelReference>
        <DMF_ContextModelReference>
         <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
         <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
         <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
         <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
        </DMF_ContextModelReference>
       </DMF_ContextModelReferences>
       <DMF_Content>
        <Dialog>
         <DialogID>"__DLG_0005_01_0001"</DialogID>
         <DialogName>"Reisende spezifizieren"</DialogName>
         <DialogDescription>""</DialogDescription>
         <DialogLabel>"Reisende spezifizieren"</DialogLabel>
         <DialogType>Single</DialogType>
         <Position>
          <Predecessors>
           <Predecessor>
            <PRED_DialogID>""</PRED_DialogID>
            <PRED_DialogName>""</PRED_DialogName>
           </Predecessor>
          </Predecessors>
          <Successors>
           <Successor>
            <SUCC_DialogID>""</SUCC_DialogID>
            <SUCC_DialogName>""</SUCC_DialogName>
            <SUCC_TransitionType>Sequential</SUCC_TransitionType>
            <SUCC_Trigger>
             <SUCC_ConceptID>""</SUCC_ConceptID>
             <SUCC_ConceptName>""</SUCC_ConceptName>
            </SUCC_Trigger>
           </Successor>
          </Successors>
         </Position>
         <DLG_Tasks>
          <DLG_Task>
           <DLG_TaskID>"__TSK_0005_01_0001"</DLG_TaskID>
           <DLG_TaskName>"Reisende spezifizieren"</DLG_TaskName>
           <DLG_Processing>Recursive</DLG_Processing>
          </DLG_Task>
         </DLG_Tasks>
        </Dialog>
       </DMF_Content>
      </MDFR_Fragment>
     </ModelFragment>
    </ModelFragments>
   </PaMGIS>
  </Deployment>
</Pattern>
```

J.6 Reiseoptionen spezifizieren

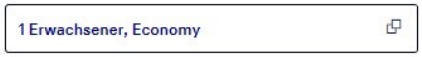
```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0006"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reiseoptionen spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0006_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind die buchbaren Reiseoptionen, wie beispielsweise die Reise-
                  klasse, festzulegen."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zu den Reiseoptionen werden benötigt:
                  1. Auswahl der Reiseklasse, beispielsweise 1. Klasse oder 2. Klasse
                     bei Bahnreisen oder Business-Class oder Economy.Class bei Flügen
                  2. Optionale Auswahl weiterer Kriterien, wie beispielsweise die Art
                     des Verkehrsmittels (Schnellzug oder Regionalzug) oder die Art
                     der Reiseverbindung (direkt oder mit Umsteigen)"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
```

```
   <KnownUses>
    <KnownUse>
     <KNWN_KnownUseID>"0006_0001"</KNWN_KnownUseID>
     <KNWN_Label>"Angaben zu gewünschten Reiseoptionen - Deutsche Bahn AG"
     </KNWN_Label>
     <KNWN_Description>"Angabe der Reiseoptionen im Online-Portal der Deutschen Bahn
                        (www.bahn.de)"</KNWN_Description>
     <KNWN_Images>
      <KNWN_Image>
```


```
      </KNWN_Image>
      <KNWN_Image>
```


```
      </KNWN_Image>
      <KNWN_Image>
```


```
      </KNWN_Image>
     </KNWN_Images>
    </KnownUse>
    <KnownUse>
     <KNWN_KnownUseID>"0006_0002"</KNWN_KnownUseID>
     <KNWN_Label>"Angaben zu gewünschten Reiseoptionen -Deutsche Lufthansa
AG"</KNWN_Label>
     <KNWN_Description>"Angabe der Reiseoptionen im Online-Portal der Lufthansa
                        (www.lufthansa.de)"</KNWN_Description>
     <KNWN_Images>
      <KNWN_Image>
```

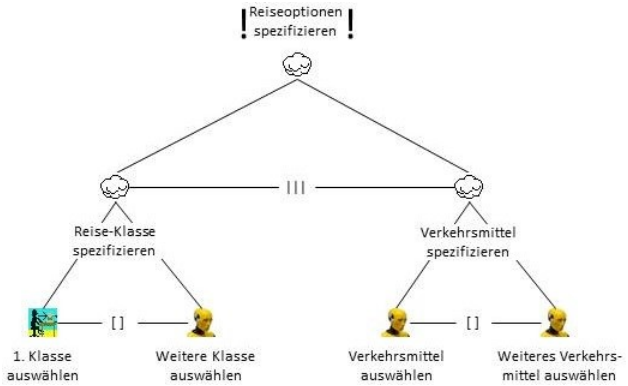



```
      </KNWN_Image>
      <KNWN_Image>
```


```
      </KNWN_Image>
     </KNWN_Images>
    </KnownUse>
   </KnownUses>
  </Practice>
 </Body>
 <Relationship>
  <RLTN_Information>""</RLTN_Information>
  <RLTN_Diagram>
```


```
  </RLTN_Diagram>
  <Relations>
   <Relation>
    <RLTN_RelationID>"RLN_0001_0006_0001"</RLTN_RelationID>
    <RLTN_Nature>Internal</RLTN_Nature>
    <RLTN_Type>Komposition</RLTN_Type>
    <RLTN_Sense>"ist Teil von"</RLTN_Sense>
    <RLTN_Reference>
     <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
```

```
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0002"</RLTN_PatternID>
    <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0006_01"</MDFR_FragmentID>
    <MDFR_Label>"Reiseoptionen spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren der Reise-Klasse und Verkehrsmittel"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>True</IncludesDummy>
     <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0006_01_0001"</TaskID>
       <TaskName>"Reiseoptionen spezifizieren"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Abstraction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reiseoptionen spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>""</ParentID>
         <ParentName>""</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>""</SiblingLeftID>
         <SiblingLeftName>""</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
```

```
</Position>
<TemporalOperator>Disabling</TemporalOperator>
<UIConcepts></UIConcepts>
<Subtasks>
 <Subtask>
  <TaskID>"__TSK_0006_01_0002"</TaskID>
  <TaskName>"Reise-Klasse spezifizieren"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
   <TaskOriginPatternName>"Reiseoptionen spezifizieren"
   </TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0006_01_0001"</ParentID>
    <ParentName>"Reiseoptionen spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0006_01_0003"<SiblingRightID>
    <SiblingRightName>"Verkehrsmittel spezifizieren"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Interleaving</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0006_01_0001"</UIConceptID>
    <UIConceptName>"Reiseklasse"</UIConceptName>
   </UIConcept>
  </UIConcepts>
  </Subtasks>
   <Subtask>
    <TaskID>"__TSK_0006_01_0004"</TaskID>
    <TaskName>"1. Klasse wählen"</TaskName>
    <TaskDescription>""</TaskDescription>
    <TaskType>Interaction</TaskType>
    <TaskOrigin>
     <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
     <TaskOriginPatternName>"Reiseoptionen spezifizieren"
     </TaskOriginPatternName>
     <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
     <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
    </TaskOrigin>
    <ContextsOfUse></ContextsOfUse>
    <Optional>False</Optional>
    <Iterative>False</Iterative>
    <Conditional>True</Conditional>
    <Preconditions></Preconditions>
    <Postconditions>
     <Postcondition>
      <Operator>"assignment"</Operator>
      <Operands>
       <Operand>
        <OperandType>"Concept"</OperandType>
        <OperandID>"__CPT_0006_01_0001"</OperandID>
        <OperandName>"Reiseklasse"</OperandName>
        <OperandElement>"CCPT_Value"</OperandElement>
       </Operand>
```

```
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"1. Klasse"</OperandValue>
    </Operand>
   </Operands>
  </Postcondition>
 </Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0006_01_0002"</ParentID>
   <ParentName>"Reise-Klasse spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>""</SiblingLeftID>
   <SiblingLeftName>""</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>"__TSK_0006_01_0005"<SiblingRightID>
   <SiblingRightName>"Weitere Klasse auswählen"</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator>Choice</TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0006_01_0003"</UIConceptID>
   <UIConceptName>"Reiseklasse 1"</UIConceptName>
  </UIConcept>
 </UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"  TSK 0006 01 0005"</TaskID>
 <TaskName>"Weitere Klasse auswählen"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Dummy</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reiseoptionen spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
  <Postcondition>
   <Operator>"assignment"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0006_01_0001"</OperandID>
     <OperandName>"Reiseklasse"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Dummy"</OperandValue>
    </Operand>
   </Operands>
  </Postcondition>
 </Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0006_01_0002"</ParentID>
   <ParentName>"Reise-Klasse spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0006_01_0004"</SiblingLeftID>
   <SiblingLeftName>"1. Klasse auswählen"</SiblingLeftName>
```

```
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Choice</TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"  CPT 0006 01 0004"</UIConceptID>
     <UIConceptName>"Weitere Reise Dummy"</UIConceptName>
    </UIConcept>
   </UIConcepts>
  </Subtask>
 </Subtasks>
</Subtask>
<Subtask>
 <TaskID>"__TSK_0006_01_0003"</TaskID>
 <TaskName>"Verkehrsmittel spezifizieren"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Abstraction</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reiseoptionen spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>False</Conditional>
 <Preconditions></Preconditions>
 <Postconditions></Postconditions>
 <Position>
  <Parent>
   <ParentID>"__TSK_0006_01_0001"</ParentID>
   <ParentName>"Reiseoptionen spezifizieren"</ParentName>
  </Parent>
  <SiblingLeft>
   <SiblingLeftID>"__TSK_0006_01_0002"</SiblingLeftID>
   <SiblingLeftName>"Reise-Klasse spezifizieren"</SiblingLeftName>
  </SiblingLeft>
  <SiblingRight>
   <SiblingRightID>""<SiblingRightID>
   <SiblingRightName>""</SiblingRightName>
  </SiblingRight>
 </Position>
 <TemporalOperator></TemporalOperator>
 <UIConcepts>
  <UIConcept>
   <UIConceptID>"__CPT_0006_01_0002"</UIConceptID>
   <UIConceptName>"Verkehrsmittel"</UIConceptName>
  </UIConcept>
 </UIConcepts>
 <Subtasks>
  <Subtask>
   <TaskID>"__TSK_0006_01_0006"</TaskID>
   <TaskName>"Verkehrsmittel auswählen"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Dummy</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
    <TaskOriginPatternName>"Reiseoptionen spezifizieren"
    </TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
```

```
<Conditional>True</Conditional>
<Preconditions></Preconditions>
<Postconditions>
 <Postcondition>
  <Operator>"assignment"</Operator>
  <Operands>
   <Operand>
    <OperandType>"Concept"</OperandType>
    <OperandID>"__CPT_0006_01_0002"</OperandID>
    <OperandName>"Verkehrsmittel"</OperandName>
    <OperandElement>"CCPT_Value"</OperandElement>
   </Operand>
   <Operand>
    <OperandType>"Constant"</OperandType>
    <OperandValue>"Dummy"</OperandValue>
   </Operand>
  </Operands>
 </Postcondition>
</Postconditions>
<Position>
 <Parent>
  <ParentID>"__TSK_0006_01_0003"</ParentID>
  <ParentName>"Verkehrsmittel spezifizieren"</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>""</SiblingLeftID>
  <SiblingLeftName>""</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>"  TSK 0006 01 0007"<SiblingRightID>
  <SiblingRightName>"Weiteres Verkehrsmittel auswählen"</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>Choice</TemporalOperator>
<UIConcepts>
 <UIConcept>
  <UIConceptID>"__CPT_0006_01_0005"</UIConceptID>
  <UIConceptName>"Verkehrsmittel 1"</UIConceptName>
 </UIConcept>
</UIConcepts>
</Subtask>
<Subtask>
 <TaskID>"  TSK 0006 01 0007"</TaskID>
 <TaskName>"Weiteres Verkehrsmittel auswählen"</TaskName>
 <TaskDescription>""</TaskDescription>
 <TaskType>Dummy</TaskType>
 <TaskOrigin>
  <TaskOriginPatternID>"PPT_0001_0006"</TaskOriginPatternID>
  <TaskOriginPatternName>"Reiseoptionen spezifizieren"
  </TaskOriginPatternName>
  <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
  <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
 </TaskOrigin>
 <ContextsOfUse></ContextsOfUse>
 <Optional>False</Optional>
 <Iterative>False</Iterative>
 <Conditional>True</Conditional>
 <Preconditions></Preconditions>
 <Postconditions>
  <Postcondition>
   <Operator>"assignment"</Operator>
   <Operands>
    <Operand>
     <OperandType>"Concept"</OperandType>
     <OperandID>"__CPT_0006_01_0002"</OperandID>
     <OperandName>"Verkehrsmittel"</OperandName>
     <OperandElement>"CCPT_Value"</OperandElement>
    </Operand>
    <Operand>
     <OperandType>"Constant"</OperandType>
     <OperandValue>"Dummy"</OperandValue>
```

```
          </Operand>
         </Operands>
        </Postcondition>
       </Postconditions>
       <Position>
        <Parent>
         <ParentID>"__TSK_0006_01_0003"</ParentID>
         <ParentName>"Verkehrsmittel spezifizieren"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0006_01_0004"</SiblingLeftID>
         <SiblingLeftName>"Verkehrsmittel auswählen"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>Choice</TemporalOperator>
       <UIConcepts>
        <UIConcept>
         <UIConceptID>"__CPT_0006_01_0006"</UIConceptID>
         <UIConceptName>"Weiteres Verkehrsmittel Dummy"</UIConceptName>
        </UIConcept>
       </UIConcepts>
      </Subtask>
     </Subtasks>
    </Subtask>
   </Subtasks>
  </Subtask>
 </TMF_Content>
</MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0006_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Reiseoptionen spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reiseoptionen spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0006_01_0001	Reiseklasse	SingleChoice
2	__CPT_0006_01_0002	Verkehrsmittel	SingleChoice
3	__CPT_0006_01_0003	Reiseklasse 1	ChoiceItem
4	__CPT_0006_01_0004	Weitere Reiseklasse Dummy	ChoiceItem
5	__CPT_0006_01_0005	Verkehrsmittel 1	SingleChoice
6	__CPT_0006_01_0006	Weiteres Verkehrsmittel Dummy	SingleChoice

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>True</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0006_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Reiseklasse"</CCPT_ConceptName>
    <CCPT_Description>"Auswahl der Reise-Klasse"</CCPT_Description>
    <CCPT_Label>"Klasse:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
```

```
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0006_01_0002"</CCPT_TaskID>
   <CCPT_TaskName>"Reise-Klasse spezifizieren"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0006_01_0002"</CCPT_ConceptID>
 <CCPT_ConceptName>"Verkehrsmittel"</CCPT_ConceptName>
 <CCPT_Description>"Auswahl der relevantenm Verkehrsmittel"</CCPT_Description>
 <CCPT_Label>"Verkehrsmittel:"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0006_01_0003"</CCPT_TaskID>
   <CCPT_TaskName>"Verkehrsmittel spezifizieren"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0006_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Reiseklasse 1"</CCPT_ConceptName>
 <CCPT_Description>"1. Klasse auswählen"</CCPT_Description>
 <CCPT_Label>"1. Klasse"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0006_01_0004"</CCPT_TaskID>
   <CCPT_TaskName>"1. Klasse auswählen"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0006_01_0004"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weitere Reiseklasse Dummy"</CCPT_ConceptName>
 <CCPT_Description>"Weitere Klasse auswählen"</CCPT_Description>
 <CCPT_Label>"Weitere Klasse"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
```

```
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0006_01_0005"</CCPT_TaskID>
   <CCPT_TaskName>"Weitere Klasse auswählen"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0006_01_0005"</CCPT_ConceptID>
 <CCPT_ConceptName>"Verkehrsmittel 1"</CCPT_ConceptName>
 <CCPT_Description>"Verkehrsmittel auswählen"</CCPT_Description>
 <CCPT_Label>"Verkehrsmittel"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0006_01_0006"</CCPT_TaskID>
   <CCPT_TaskName>"Verkehrsmittel auswählen"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0006_01_0006"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weiteres Verkehrsmittel Dummy"</CCPT_ConceptName>
 <CCPT_Description>"Weiteres Verkehrsmittel auswählen"</CCPT_Description>
 <CCPT_Label>"Weiteres Verkehrsmittel"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>False</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0006"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseoptionen spezifizieren"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
```

```
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"  TSK_0006_01_0007"</CCPT_TaskID>
      <CCPT_TaskName>"Weiteres Verkehrsmittel auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0006_01"</MDFR_FragmentID>
 <MDFR_Label>"Reiseoptionen spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Festlegen der Reiseoptionen"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```

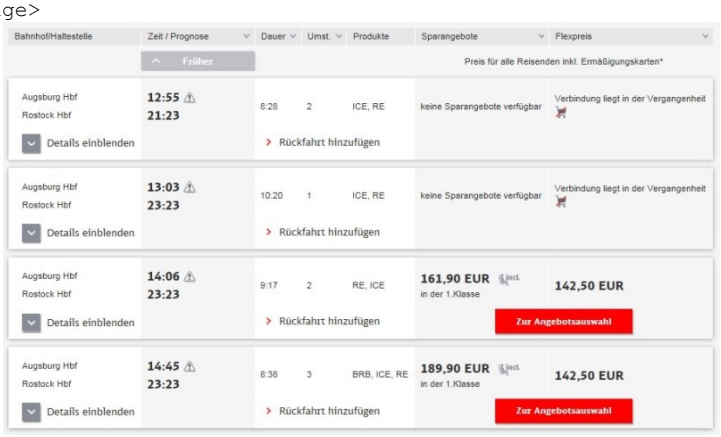

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0006_01_0001"</DialogID>
    <DialogName>"Reiseoptionen spezifizieren"</DialogName>
    <DialogDescription>""</DialogDescription>
    <DialogLabel>"Reiseoptionen spezifizieren"</DialogLabel>
    <DialogType>Single</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED_DialogID>""</PRED_DialogID>
       <PRED_DialogName>""</PRED_DialogName>
      </Predecessor>
     </Predecessors>
     <Successors>
      <Successor>
       <SUCC_DialogID>""</SUCC_DialogID>
       <SUCC_DialogName>""</SUCC_DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>""</SUCC_ConceptID>
        <SUCC_ConceptName>""</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0006_01_0001"</DLG_TaskID>
      <DLG_TaskName>"Reiseoptionen spezifizieren"</DLG_TaskName>
      <DLG_Processing>Recursive</DLG_Processing>
     </DLG_Task>
    </DLG_Tasks>
```

```
        </Dialog>
       </DMF_Content>
      </MDFR_Fragment>
     </ModelFragment>
    </ModelFragments>
   </PaMGIS>
  </Deployment>
 </Pattern>
```

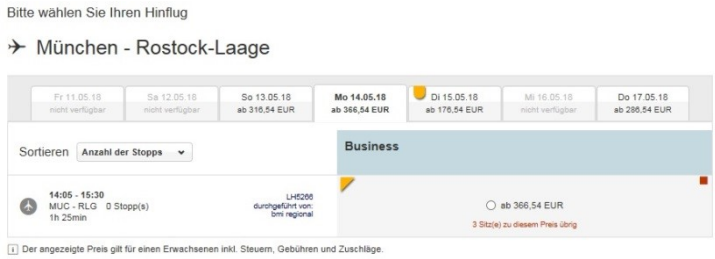
J.7 Reiseverbindung wählen

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0007"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Reiseverbindung wählen"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"14.05.2018"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0007_0001"</CHNG_ChangeID>
     <CHNG_Date>"14.05.2018"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Die gewünschte Reiseverbindung ist festzulegen."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Auswahlen werden benötigt:
                  1. Auswahl der gewünschten Hinreise aus seiner Menge von Reisevor-
                     schlägen, die die zuvor angegebenen Hinreise-Kriterien erfüllen
                  2. Optionale Auswahl der gewünschten Rückreise aus seiner Menge von
                     Reisevorschlägen, die die zuvor angegebenen Rückreise-Kriterien
                     erfüllen"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
   <KnownUses>
```

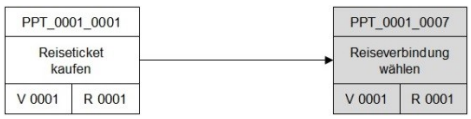
```
   <KnownUse>
    <KNWN_KnownUseID>"0007_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Auswahl der Reiseverbindungen – Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Auswahl der Reiseverbindungen im Online-Portal der Deutschen
                       Bahn (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```




```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
   <KnownUse>
    <KNWN_KnownUseID>"0007_0002"</KNWN_KnownUseID>
    <KNWN_Label>"Auswahl der Reiseverbindungen – Deutsche Lufthansa AG"</KNWN_Label>
    <KNWN_Description>"Auswahl der Reiseverbindungen im Online-Portal der Lufthansa
                       (www.lufthansa.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```

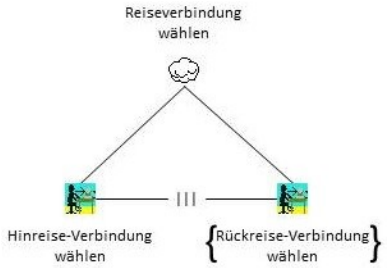



```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0007_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
    <RLTN_PatternName>"Reiseticket kaufen"</RLTN_PatternName>
```

```
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0007_01"</MDFR_FragmentID>
    <MDFR_Label>"Reiseverbindung wählen"</MDFR_Label>
    <MDFR_Purpose>"Auswählen der Verbindungen für die Hin- und Rückreise"
    </MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>False</IncludesDummy>
     <TMF_ProposedTempOp>SequentialEnabling</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0007_01_0001"</TaskID>
       <TaskName>"Reiseverbindung wählen"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Abstraction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0007"</TaskOriginPatternID>
        <TaskOriginPatternName>"Reiseverbindung wählen"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>""</ParentID>
         <ParentName>""</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>""</SiblingLeftID>
         <SiblingLeftName>""</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>SequentialEnabling</TemporalOperator>
       <UIConcepts></UIConcepts>
       <Subtasks>
        <Subtask>
         <TaskID>"__TSK_0007_01_0002"</TaskID>
         <TaskName>"Hinreise-Verbindung wählen"</TaskName>
         <TaskDescription>""</TaskDescription>
```

```
<TaskType>Interaction</TaskType>
<TaskOrigin>
 <TaskOriginPatternID>"PPT_0001_0007"</TaskOriginPatternID>
 <TaskOriginPatternName>"Reiseverbindung wählen"</TaskOriginPatternName>
 <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
 <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
</TaskOrigin>
<ContextsOfUse></ContextsOfUse>
<Optional>False</Optional>
<Iterative>False</Iterative>
<Conditional>False</Conditional>
<Preconditions></Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
  <ParentID>"__TSK_0007_01_0001"</ParentID>
  <ParentName>"Reiseverbindung wählen"</ParentName>
 </Parent>
 <SiblingLeft>
  <SiblingLeftID>""</SiblingLeftID>
  <SiblingLeftName>""</SiblingLeftName>
 </SiblingLeft>
 <SiblingRight>
  <SiblingRightID>"__TSK_0007_01_0003"<SiblingRightID>
  <SiblingRightName>"Rückreise-Verbindung wählen"</SiblingRightName>
 </SiblingRight>
</Position>
<TemporalOperator>Interleaving</TemporalOperator>
<UIConcepts>
 <UIConcept>
  <UIConceptID>"__CPT_0007_01_0001"</UIConceptID>
  <UIConceptName>"Hinreiseverbindung"</UIConceptName>
 </UIConcept>
</UIConcepts>
</Subtask>
<Subtask>
<TaskID>"__TSK_0007_01_0003"</TaskID>
<TaskName>"Rückreise-Verbindung wählen"</TaskName>
<TaskDescription>""</TaskDescription>
<TaskType>Interaction</TaskType>
<TaskOrigin>
 <TaskOriginPatternID>"PPT_0001_0007"</TaskOriginPatternID>
 <TaskOriginPatternName>"Reiseverbindung wählen"
 </TaskOriginPatternName>
 <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
 <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
</TaskOrigin>
<ContextsOfUse></ContextsOfUse>
<Optional>False</Optional>
<Iterative>False</Iterative>
<Conditional>True</Conditional>
<Preconditions>
 <Precondition>
  <Operator>"equal"</Operator>
  <Operands>
   <Operand>
    <OperandType>"Concept"</OperandType>
    <OperandID>"__CPT_0003_01_0004"</OperandID>
    <OperandName>"Reiserichtung"</OperandName>
    <OperandElement>"CCPT_Value"</OperandElement>
   </Operand>
   <Operand>
    <OperandType>"Constant"</OperandType>
    <OperandValue>"Hin- und Rückreise"</OperandValue>
   </Operand>
  </Operands>
 </Precondition>
</Preconditions>
<Postconditions></Postconditions>
<Position>
 <Parent>
```

```
        <ParentID>"__TSK_0007_01_0001"</ParentID>
        <ParentName>"Reiseverbindung wählen"</ParentName>
       </Parent>
       <SiblingLeft>
        <SiblingLeftID>"__TSK_0007_01_0002"</SiblingLeftID>
        <SiblingLeftName>"Hinreise-Verbindung wählen"</SiblingLeftName>
       </SiblingLeft>
       <SiblingRight>
        <SiblingRightID>""<SiblingRightID>
        <SiblingRightName>""</SiblingRightName>
       </SiblingRight>
      </Position>
      <TemporalOperator></TemporalOperator>
      <UIConcepts>
       <UIConcept>
        <UIConceptID>"__CPT_0007_01_0002"</UIConceptID>
        <UIConceptName>"Rückreiseverbindung"</UIConceptName>
       </UIConcept>
      </UIConcepts>
     </Subtask>
    </Subtasks>
   </Subtask>
  </TMF_Content>
 </MDFR_Fragment>
</ModelFragment>
</ModelFragments>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0007_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte – Reiseverbindung wählen"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Reiseverbindung wählen“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0007_01_0001	Hinreiseverbindung	SingleChoice
2	__CPT_0007_01_0002	Rückreiseverbindung	SingleChoice
3	__CPT_0007_01_0003	Weiter zum Folgedialog (4)	Navigator
4	__CPT_0007_01_0004	Weiter zum Folgedialog (5)	Navigator

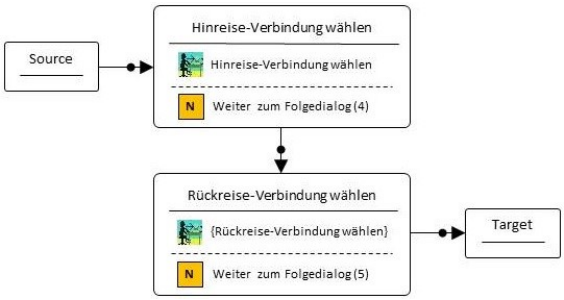
```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>False</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0007_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Hinreiseverbindung"</CCPT_ConceptName>
    <CCPT_Description>"Auswahl der Hinreiseverbindung"</CCPT_Description>
    <CCPT_Label>"Verbindungen Hinreise"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0007"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Reiseverbindung wählen"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0007_01_0002"</CCPT_TaskID>
      <CCPT_TaskName>"Hinreise-Verbindung wählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
```

```
<Concept>
 <CCPT_ConceptID>"__CPT_0007_01_0002"</CCPT_ConceptID>
 <CCPT_ConceptName>"Rückreiseverbindung"</CCPT_ConceptName>
 <CCPT_Description>"Auswahl der Rückreiseverbindung"</CCPT_Description>
 <CCPT_Label>"Verbindungen Rückreise"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
 <CCPT_DataType>"String"</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0007"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseverbindung wählen"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks>
  <CCPT_TaskLink>
   <CCPT_TaskID>"__TSK_0007_01_0003"</CCPT_TaskID>
   <CCPT_TaskName>"Rückreise-Verbindung wählen"</CCPT_TaskName>
  </CCPT_TaskLink>
 </CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0007_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weiter zum Folgedialog (4)"</CCPT_ConceptName>
 <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                   modellierung im Dialog-Modell-Fragment benötigt wird"
 </CCPT_Description>
 <CCPT_Label>"Weiter"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Navigator</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0007"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseverbindung wählen"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks></CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0007_01_0004"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weiter zum Folgedialog (5)"</CCPT_ConceptName>
 <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                   modellierung im Dialog-Modell-Fragment benötigt wird"
 </CCPT_Description>
 <CCPT_Label>"Weiter"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Navigator</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0007"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Reiseverbindung wählen"</CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
```

```
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks></CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0007_01"</MDFR_FragmentID>
 <MDFR_Label>"Reiseverbindung auswählen"</MDFR_Label>
 <MDFR_Purpose>"Auswahl der Hin- und Rückreiseverbindungen"</MDFR_Purpose>
 <MDFR_Annotation>"Für den Kontext - Großer Bildschirm -"</MDFR_Annotation>
 <MDFR_Diagram>
```


```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0004_01_0001"</DialogID>
    <DialogName>"Reiseverbindung wählen"</DialogName>
    <DialogDescription>""</DialogDescription>
    <DialogLabel>"Reiseverbindungen wählen"</DialogLabel>
    <DialogType>Modal</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED_DialogID>""</PRED_DialogID>
       <PRED_DialogName>""</PRED_DialogName>
      </Predecessor>
     </Predecessors>
     <Successors>
      <Successor>
       <SUCC_DialogID>""</SUCC_DialogID>
       <SUCC_DialogName>""</SUCC_DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>"__CPT_0007_01_0003"</SUCC_ConceptID>
        <SUCC_ConceptName>"Weiter zum Folgedialog (4)"</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0007_01_0002"</DLG_TaskID>
      <DLG_TaskName>"Hinreise-Verbindung wählen"</DLG_TaskName>
      <DLG_Processing>Exclusive</DLG_Processing>
     </DLG_Task>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0007_01_0003"</DLG_TaskID>
      <DLG_TaskName>"Rückreise-Verbindung wählen"</DLG_TaskName>
      <DLG_Processing>Exclusive</DLG_Processing>
     </DLG_Task>
    </DLG_Tasks>
    <SupplementaryConcepts>
```

```
     <SupplementaryConcept>
      <SupplementaryConceptID>"__CPT_0007_01_0003"</SupplementaryConceptID>
      <SupplementaryConceptName>"Weiter zum Folgedialog (4)"
      </SupplementaryConceptName>
     </SupplementaryConcept>
    </SupplementaryConcepts>
   </Dialog>
  </DMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0007_02"</MDFR_FragmentID>
 <MDFR_Label>"Reiseverbindung auswählen"</MDFR_Label>
 <MDFR_Purpose>"Auswahl der Hin- und Rückreiseverbindungen"</MDFR_Purpose>
 <MDFR_Annotation>"Für den Kontext - Kleiner Bildschirm -"</MDFR_Annotation>
 <MDFR_Diagram>
```



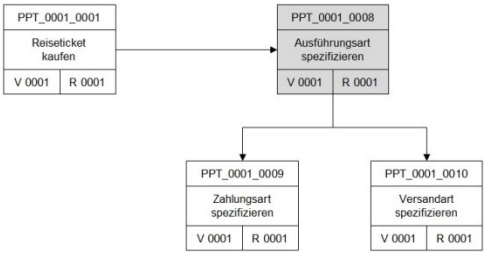

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0007_02_0001"</DialogID>
    <DialogName>"Hinreise-Verbindung wählen"</DialogName>
    <DialogDescription>""</DialogDescription>
    <DialogLabel>"Hinreiseverbindung wählen"</DialogLabel>
    <DialogType>Modal</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED_DialogID>""</PRED_DialogID>
       <PRED_DialogName>""</PRED_DialogName>
      </Predecessor>
     </Predecessors>
     <Successors>
      <Successor>
       <SUCC_DialogID>"__DLG_0007_02_0002"</SUCC_DialogID>
       <SUCC_DialogName>"Rückreise-Verbindung wählen"</SUCC_DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>"__CPT_0007_01_0003"</SUCC_ConceptID>
        <SUCC_ConceptName>"Weiter zum Folgedialog (4)"</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks>
     <DLG_Task>
      <DLG_TaskID>"__TSK_0007_01_0002"</DLG_TaskID>
      <DLG_TaskName>"Hinreise-Verbindung wählen"</DLG_TaskName>
      <DLG_Processing>Exclusive</DLG_Processing>
     </DLG_Task>
```

```
        </DLG_Tasks>
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0007_01_0003"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog (4)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
       <Dialog>
        <DialogID>"__DLG_0007_02_0002"</DialogID>
        <DialogName>"Rückreise-Verbindung wählen"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Rückreiseverbindung wählen"</DialogLabel>
        <DialogType>Modal</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>"__DLG_0007_02_0001"</PRED_DialogID>
           <PRED_DialogName>"Hinreise-Verbindung wählen"</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>"__CPT_0007_01_0004"</SUCC_ConceptID>
            <SUCC_ConceptName>"Weiter zum Folgedialog (5)"</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0007_01_0003"</DLG_TaskID>
          <DLG_TaskName>"Rückreise-Verbindung wählen"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0007_01_0004"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog (5)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.8 Ausführungsart spezifizieren

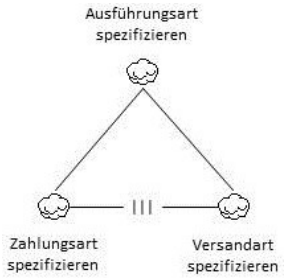
```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0008"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Ausführungsart spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0008_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sind diejenigen Daten festzulegen, die für die Ausführung des
                  Ticketkaufs erforderlich sind."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Angaben zur Ausführung des Kaufs werden benötigt:
                  1. Es muss die Zahlungsart festgelegt werden
                  2. Es muss die Art der Übergabe des Tickets an den Käufer festgelegt
                     werden"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
   <KnownUses>
    <KnownUse>
```

```
    <KNWN_KnownUseID>"0008_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Auswahl der Ausführungsart – Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Beispiele für die Wahl der Zahlungsart und der Versandart
                       siehe Spezifikationen von Pattern PPT_0001_0009 und Pattern
                       PPT_0001_0010."</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image></KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0008_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
    <RLTN_PatternName>"Reiseticket kaufen"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0008_0002"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"besteht aus"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0009"</RLTN_PatternID>
    <RLTN_PatternName>"Zahlungsart spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0008_0003"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"besteht aus"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0010"</RLTN_PatternID>
    <RLTN_PatternName>"Versandart spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
```

```
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0008_01"</MDFR_FragmentID>
    <MDFR_Label>"Ausführungsart spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren der Zahlungs- und der Versandart für die Reise-
                  tickets"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <TMF_IncludesDummy>False</IncludesDummy>
     <TMF_ProposedTempOp>SequentialEnabling</TMF_ProposedTempOp>
     <TMF_Content>
      <Subtask>
       <TaskID>"__TSK_0008_01_0001"</TaskID>
       <TaskName>"Ausführungsart spezifizieren"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Abstraction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0008"</TaskOriginPatternID>
        <TaskOriginPatternName>"Ausführungsart spezifizieren"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>""</ParentID>
         <ParentName>""</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>""</SiblingLeftID>
         <SiblingLeftName>""</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>SequentialEnabling</TemporalOperator>
       <UIConcepts></UIConcepts>
       <Subtasks>
        <Subtask>
         <TaskID>"__TSK_0008_01_0002"</TaskID>
         <TaskName>"Zahlungsart spezifizieren"</TaskName>
         <TaskDescription>""</TaskDescription>
         <TaskType>Abstraction</TaskType>
         <TaskOrigin>
          <TaskOriginPatternID>"PPT_0001_0008"</TaskOriginPatternID>
```

```
      <TaskOriginPatternName>"Ausführungsart spezifizieren"
      </TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>False</Conditional>
     <Preconditions></Preconditions>
     <Postconditions></Postconditions>
     <Position>
      <Parent>
       <ParentID>"__TSK_0008_01_0001"</ParentID>
       <ParentName>"Ausführungsart spezifizieren"</ParentName>
      </Parent>
      <SiblingLeft>
       <SiblingLeftID>""</SiblingLeftID>
       <SiblingLeftName>""</SiblingLeftName>
      </SiblingLeft>
      <SiblingRight>
       <SiblingRightID>"__TSK_0008_01_0003"<SiblingRightID>
       <SiblingRightName>"Versandart spezifizieren"</SiblingRightName>
      </SiblingRight>
     </Position>
     <TemporalOperator>Interleaving</TemporalOperator>
     <UIConcepts></UIConcepts>
    </Subtask>
    <Subtask>
     <TaskID>"__TSK_0008_01_0003"</TaskID>
     <TaskName>"Versandart spezifizieren"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Abstraction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0008"</TaskOriginPatternID>
      <TaskOriginPatternName>"Ausführungsart spezifizieren"
      </TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>False</Conditional>
     <Preconditions></Preconditions>
     <Postconditions></Postconditions>
     <Position>
      <Parent>
       <ParentID>"__TSK_0008_01_0001"</ParentID>
       <ParentName>"Ausführungsart spezifizieren"</ParentName>
      </Parent>
      <SiblingLeft>
       <SiblingLeftID>"__TSK_0008_01_0002"</SiblingLeftID>
       <SiblingLeftName>"Zahlungsart spezifizieren"</SiblingLeftName>
      </SiblingLeft>
      <SiblingRight>
       <SiblingRightID>""<SiblingRightID>
       <SiblingRightName>""</SiblingRightName>
      </SiblingRight>
     </Position>
     <TemporalOperator></TemporalOperator>
     <UIConcepts></UIConcepts>
    </Subtask>
   </Subtasks>
  </Subtask>
 </TMF_Content>
</MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0008_01"</MDFR_FragmentID>
```

```
<MDFR_Label>"Konzepte – Ausführungsart spezifizieren"</MDFR_Label>
<MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
</MDFR_Purpose>
<MDFR_Annotation>""</MDFR_Annotation>
<MDFR_Diagram>
```

Konzepte des Patterns „Ausführungsart spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0008_01_0001	Weiter zum Folgedialog (6)	Navigator

```
</MDFR_Diagram>
<MDFR_Fragment>
 <CMF_IncludesDummy>False</CMF_IncludesDummy>
 <CMF_Content>
  <Concept>
   <CCPT_ConceptID>"__CPT_0008_01_0001"</CCPT_ConceptID>
   <CCPT_ConceptName>"Weiter zum Folgedialog (6)"</CCPT_ConceptName>
   <CCPT_Description>"Zusätzliches Konzept, das zur Navigation bei der Dialog-
                      modellierung im Dialog-Modell-Fragment benötigt wird"
   </CCPT_Description>
   <CCPT_Label>"Weiter"</CCPT_Label>
   <CCPT_Perceptible>True</CCPT_Perceptible>
   <CCPT_Enabled>True</CCPT_Enabled>
   <CCPT_Required>True</CCPT_Required>
   <CCPT_ConceptType>Navigator</CCPT_ConceptType>
   <CCPT_DataType>""</CCPT_DataType>
   <CCPT_Value>""</CCPT_Value>
   <CCPT_ConceptOrigin>
    <CCPT_OriginPatternID>"PPT_0001_0008"</CCPT_OriginPatternID>
    <CCPT_OriginPatternName>"Ausführungsart spezifizieren"
    </CCPT_OriginPatternName>
    <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
    <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
   </CCPT_ConceptOrigin>
   <CCPT_Preconditions></CCPT_Preconditions>
   <CCPT_Postconditions><CCPT_Postconditions>
   <CCPT_DataLink></CCPT_DataLink>
   <CCPT_TaskLinks></CCPT_TaskLinks>
  </Concept>
 </CMF_Content>
</MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0008_01"</MDFR_FragmentID>
 <MDFR_Label>"Ausführungsart spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifizieren der Ausführungsart"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```

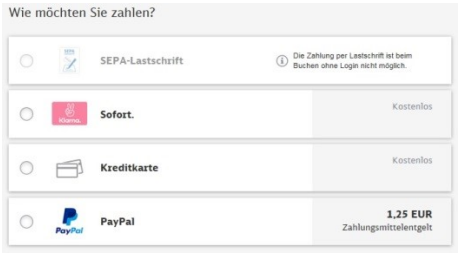

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
```

```
      <DMF_Content>
       <Dialog>
        <DialogID>"__DLG_0008_01_0001"</DialogID>
        <DialogName>"Ausführungsart spezifizieren"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Ausführungsart spezifizieren"</DialogLabel>
        <DialogType>Complex</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>""</PRED_DialogID>
           <PRED_DialogName>""</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>"__CPT_0008_01_0001"</SUCC_ConceptID>
            <SUCC_ConceptName>"Weiter zum Folgedialog (6)"</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0008_01_0002"</DLG_TaskID>
          <DLG_TaskName>"Zahlungsart spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0008_01_0003"</DLG_TaskID>
          <DLG_TaskName>"Versandart spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0008_01_0001"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog (6)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.9 Zahlungsart spezifizieren

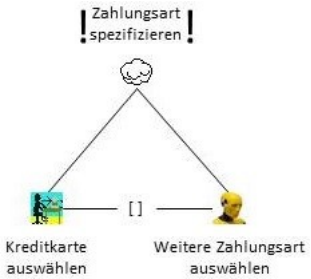
```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0009"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Zahlungsart spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0009_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es ist die Zahlungsart für den Ticketkauf festzulegen."
    </PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Auswahlmöglichkeiten hinsichtlich der Zahlungsart werden
                  benötigt:
                  1. Zahlung mittels Kreditkarte
                  2. Weitere Zahlungsarten, wie beispielsweise Rechnung, EC-Karte,
                     Kundenkarte oder Barzahlung (am Ticketautomaten)"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
   <KnownUses>
```

```
   <KnownUse>
    <KNWN_KnownUseID>"0009_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Auswahl der Zahlungsart - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Auswahl der Zahlungsart im Online-Portal der Deutschen Bahn
                       (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```




```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```


```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0009_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0008"</RLTN_PatternID>
    <RLTN_PatternName>"Ausführungsart spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0009_01"</MDFR_FragmentID>
    <MDFR_Label>"Zahlungsart spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren der Zahlungsart für die Reisetickets"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```



```
</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>True</IncludesDummy>
 <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0009_01_0001"</TaskID>
   <TaskName>"Zahlungsart spezifizieren"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0009"</TaskOriginPatternID>
    <TaskOriginPatternName>"Zahlungsart spezifizieren"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Interleaving</TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"__CPT_0009_01_0001"</UIConceptID>
     <UIConceptName>"Zahlungsart"</UIConceptName>
    </UIConcept>
   </UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0009_01_0002"</TaskID>
     <TaskName>"Kreditkarte auswählen"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Interaction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0009"</TaskOriginPatternID>
      <TaskOriginPatternName>"Zahlungsart spezifizieren"</TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>True</Conditional>
     <Preconditions></Preconditions>
     <Postconditions>
      <Postcondition>
       <Operator>"assignment"</Operator>
       <Operands>
        <Operand>
         <OperandType>"Concept"</OperandType>
         <OperandID>"__CPT_0009_01_0001"</OperandID>
         <OperandName>"Zahlungsart"</OperandName>
         <OperandElement>"CCPT_Value"</OperandElement>
        </Operand>
        <Operand>
```

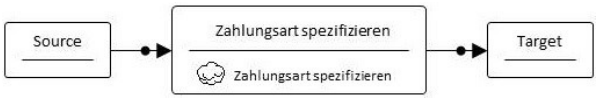
```
      <OperandType>"Constant"</OperandType>
      <OperandValue>"Kreditkarte"</OperandValue>
     </Operand>
    </Operands>
   </Postcondition>
  </Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0009_01_0001"</ParentID>
    <ParentName>"Zahlungsart spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0009_01_0003"<SiblingRightID>
    <SiblingRightName>"Weitere Zahlungsart auswählen"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Choice</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0009_01_0002"</UIConceptID>
    <UIConceptName>"Zahlungsart 1"</UIConceptName>
   </UIConcept>
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"  TSK 0009 01 0003"</TaskID>
  <TaskName>"Weitere Zahlungsart auswählen"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Dummy</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0009"</TaskOriginPatternID>
   <TaskOriginPatternName>"Zahlungsart spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>True</Conditional>
  <Preconditions></Preconditions>
  <Postconditions>
   <Postcondition>
    <Operator>"assignment"</Operator>
    <Operands>
     <Operand>
      <OperandType>"Concept"</OperandType>
      <OperandID>"__CPT_0009_01_0001"</OperandID>
      <OperandName>"Zahlungsart"</OperandName>
      <OperandElement>"CCPT_Value"</OperandElement>
     </Operand>
     <Operand>
      <OperandType>"Constant"</OperandType>
      <OperandValue>"Dummy"</OperandValue>
     </Operand>
    </Operands>
   </Postcondition>
  </Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0009_01_0001"</ParentID>
    <ParentName>"Zahlungsart spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>"__TSK_0009_01_0002"</SiblingLeftID>
    <SiblingLeftName>"Kreditkarte auswählen"</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
```

```
        <SiblingRightID>""<SiblingRightID>
        <SiblingRightName>""</SiblingRightName>
       </SiblingRight>
      </Position>
      <TemporalOperator>Choice</TemporalOperator>
      <UIConcepts>
       <UIConcept>
        <UIConceptID>"__CPT_0009_01_0003"</UIConceptID>
        <UIConceptName>"Weitere Zahlungsart Dummy"</UIConceptName>
       </UIConcept>
      </UIConcepts>
     </Subtask>
    </Subtasks>
   </Subtask>
  </TMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0009_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Zahlungsart spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Zahlungsart spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0009_01_0001	Zahlungsart	SingleChoice
2	__CPT_0009_01_0002	Zahlungsart 1	ChoiceItem
3	__CPT_0009_01_0003	Weitere Zahlungsart Dummy	ChoiceItem

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>True</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0009_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Zahlungsart"</CCPT_ConceptName>
    <CCPT_Description>"Auswahl der Zahlungsart"</CCPT_Description>
    <CCPT_Label>"Zahlungsart:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0009"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Zahlungsart spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0009_01_0001"</CCPT_TaskID>
      <CCPT_TaskName>"Zahlungsart spezifizieren"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0009_01_0002"</CCPT_ConceptID>
    <CCPT_ConceptName>"Zahlungsart 1"</CCPT_ConceptName>
    <CCPT_Description>"Kreditkarte auswählen"</CCPT_Description>
    <CCPT_Label>"Kreditkarte"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
```

```
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0009"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Zahlungsart spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0009_01_0002"</CCPT_TaskID>
      <CCPT_TaskName>"Kreditkarte auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0009_01_0003"</CCPT_ConceptID>
    <CCPT_ConceptName>"Weitere Zahlungsart Dummy"</CCPT_ConceptName>
    <CCPT_Description>"Weitere Zahlungsart auswählen"</CCPT_Description>
    <CCPT_Label>"Weitere Zahlungsart"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>Dummy</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0009"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Zahlungsart spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0009_01_0003"</CCPT_TaskID>
      <CCPT_TaskName>"Weitere Zahlungsart auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0009_01"</MDFR_FragmentID>
 <MDFR_Label>"Zahlungsart spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifizieren der Zahlungsart"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```



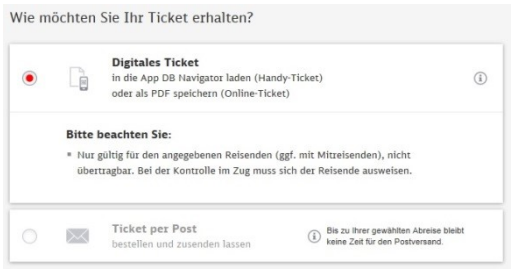

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
```

```
        <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
        <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
        <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
       </DMF_ContextModelReference>
      </DMF_ContextModelReferences>
      <DMF_Content>
       <Dialog>
        <DialogID>"__DLG_0009_01_0001"</DialogID>
        <DialogName>"Zahlungsart spezifizieren"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Zahlungsart"</DialogLabel>
        <DialogType>Single</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>""</PRED_DialogID>
           <PRED_DialogName>""</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>""</SUCC_ConceptID>
            <SUCC_ConceptName>""</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0009_01_0001"</DLG_TaskID>
          <DLG_TaskName>"Zahlungsart spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

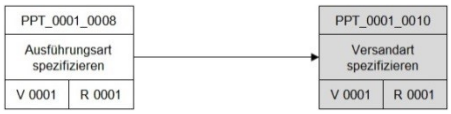
J.10 Versandart spezifizieren

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0010"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Versandart spezifizieren"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0010_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es ist die Versandart für die Tickets festzulegen."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Folgende Auswahlmöglichkeiten hinsichtlich der Versandart werden
                  benötigt:
                  1. Ausdrucken der Tickets an einem verfügbaren Drucker
                  2. Weitere Versandarten, wie beispielsweise Post-, Email- oder SMS-
                     Versand"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
   <KnownUses>
    <KnownUse>
```

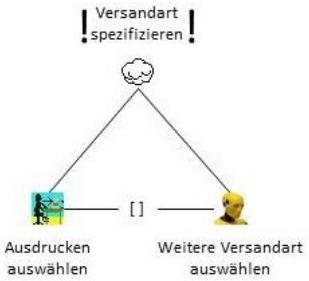
```
    <KNWN_KnownUseID>"0010_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Auswahl der Versandart - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Auswahl der Zahlungsart im Online-Portal der Deutschen Bahn
                       (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```




```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0010_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0008"</RLTN_PatternID>
    <RLTN_PatternName>"Ausführungsart spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0010_01"</MDFR_FragmentID>
    <MDFR_Label>"Versandart spezifizieren"</MDFR_Label>
    <MDFR_Purpose>"Spezifizieren der Versandart für die Reisetickets"</MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```



```
</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>True</IncludesDummy>
 <TMF_ProposedTempOp>Interleaving</TMF_ProposedTempOp>
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0010_01_0001"</TaskID>
   <TaskName>"Versandart spezifizieren"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0010"</TaskOriginPatternID>
    <TaskOriginPatternName>"Versandart spezifizieren"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Interleaving</TemporalOperator>
   <UIConcepts>
    <UIConcept>
     <UIConceptID>"__CPT_0010_01_0001"</UIConceptID>
     <UIConceptName>"Versandart"</UIConceptName>
    </UIConcept>
   </UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0010_01_0002"</TaskID>
     <TaskName>"Ausdrucken auswählen"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Interaction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0010"</TaskOriginPatternID>
      <TaskOriginPatternName>"Versandart spezifizieren"</TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>True</Conditional>
     <Preconditions></Preconditions>
     <Postconditions>
      <Postcondition>
       <Operator>"assignment"</Operator>
       <Operands>
        <Operand>
         <OperandType>"Concept"</OperandType>
         <OperandID>"__CPT_0010_01_0001"</OperandID>
         <OperandName>"Versandart"</OperandName>
         <OperandElement>"CCPT_Value"</OperandElement>
        </Operand>
        <Operand>
```

```
      <OperandType>"Constant"</OperandType>
      <OperandValue>"Ausdrucken"</OperandValue>
     </Operand>
    </Operands>
   </Postcondition>
  </Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0010_01_0001"</ParentID>
    <ParentName>"Versandart spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>"__TSK_0010_01_0003"<SiblingRightID>
    <SiblingRightName>"Weitere Versandart auswählen"</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Choice</TemporalOperator>
  <UIConcepts>
   <UIConcept>
    <UIConceptID>"__CPT_0010_01_0002"</UIConceptID>
    <UIConceptName>"Versandart 1"</UIConceptName>
   </UIConcept>
  </UIConcepts>
 </Subtask>
 <Subtask>
  <TaskID>"  TSK_0010_01_0003"</TaskID>
  <TaskName>"Weitere Versandart auswählen"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Dummy</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0010"</TaskOriginPatternID>
   <TaskOriginPatternName>"Versandart spezifizieren"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>True</Conditional>
  <Preconditions></Preconditions>
  <Postconditions>
   <Postcondition>
    <Operator>"assignment"</Operator>
    <Operands>
     <Operand>
      <OperandType>"Concept"</OperandType>
      <OperandID>"__CPT_0010_01_0001"</OperandID>
      <OperandName>"Versandart"</OperandName>
      <OperandElement>"CCPT_Value"</OperandElement>
     </Operand>
     <Operand>
      <OperandType>"Constant"</OperandType>
      <OperandValue>"Dummy"</OperandValue>
     </Operand>
    </Operands>
   </Postcondition>
  </Postconditions>
  <Position>
   <Parent>
    <ParentID>"__TSK_0010_01_0001"</ParentID>
    <ParentName>"Versandart spezifizieren"</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>"__TSK_0010_01_0002"</SiblingLeftID>
    <SiblingLeftName>"Ausdrucken auswählen"</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
```

```
        <SiblingRightID>""<SiblingRightID>
        <SiblingRightName>""</SiblingRightName>
       </SiblingRight>
      </Position>
      <TemporalOperator>Choice</TemporalOperator>
      <UIConcepts>
       <UIConcept>
        <UIConceptID>"__CPT_0010_01_0003"</UIConceptID>
        <UIConceptName>"Weitere Versandart Dummy"</UIConceptName>
       </UIConcept>
      </UIConcepts>
     </Subtask>
    </Subtasks>
   </Subtask>
  </TMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Concept</MDFR_Type>
 <MDFR_FragmentID>"__CMF_0010_01"</MDFR_FragmentID>
 <MDFR_Label>"Konzepte - Versandart spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
 </MDFR_Purpose>
 <MDFR_Annotation>""</MDFR_Annotation>
 <MDFR_Diagram>
```

Konzepte des Patterns „Versandart spezifizieren“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0010_01_0001	Versandart	SingleChoice
2	__CPT_0010_01_0002	Versandart 1	ChoiceItem
3	__CPT_0010_01_0003	Weitere Versandart Dummy	ChoiceItem

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>True</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0010_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Versandart"</CCPT_ConceptName>
    <CCPT_Description>"Auswahl der Versandart"</CCPT_Description>
    <CCPT_Label>"Versandart:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>SingleChoice</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0010"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Versandart spezifizieren"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0010_01_0001"</CCPT_TaskID>
      <CCPT_TaskName>"Versandart spezifizieren"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0010_01_0002"</CCPT_ConceptID>
    <CCPT_ConceptName>"Versandart 1"</CCPT_ConceptName>
    <CCPT_Description>"Ausdrucken auswählen"</CCPT_Description>
    <CCPT_Label>"Ausdrucken"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>ChoiceItem</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
```

```
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0010"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Versandart spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0010_01_0002"</CCPT_TaskID>
      <CCPT_TaskName>"Ausdrucken auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT ConceptID>"  CPT 0010 01 0003"</CCPT ConceptID>
    <CCPT ConceptName>"Weitere Versandart Dummy"</CCPT ConceptName>
    <CCPT_Description>"Weitere Versandart auswählen"</CCPT_Description>
    <CCPT_Label>"Weitere Versandart"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>False</CCPT_Required>
    <CCPT_ConceptType>Dummy</CCPT_ConceptType>
    <CCPT_DataType>"String"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0010"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Versandart spezifizieren"
     </CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT TaskID>"  TSK 0010 01 0003"</CCPT TaskID>
      <CCPT_TaskName>"Weitere Versandart auswählen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0010_01"</MDFR_FragmentID>
 <MDFR_Label>"Versandart spezifizieren"</MDFR_Label>
 <MDFR_Purpose>"Spezifizieren der Versandart"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```


```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
```

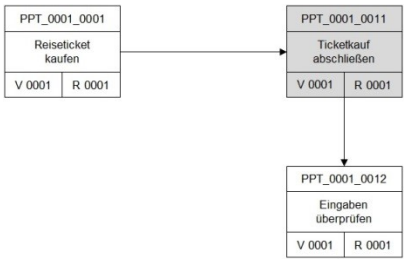
```
        <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
        <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
        <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
        <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
       </DMF_ContextModelReference>
      </DMF_ContextModelReferences>
      <DMF_Content>
       <Dialog>
        <DialogID>"__DLG_0010_01_0001"</DialogID>
        <DialogName>"Versandart spezifizieren"</DialogName>
        <DialogDescription>""</DialogDescription>
        <DialogLabel>"Versandart"</DialogLabel>
        <DialogType>Single</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>""</PRED_DialogID>
           <PRED_DialogName>""</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>""</SUCC_ConceptID>
            <SUCC_ConceptName>""</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0010_01_0001"</DLG_TaskID>
          <DLG_TaskName>"Versandart spezifizieren"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.11 Ticketkauf abschließen

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0011"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Ticketkauf abschließen"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"07.01.2017"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0011_0001"</CHNG_ChangeID>
     <CHNG_Date>"07.01.2017"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Es sollen diejenigen Aktionen durchgeführt werden, die für den
                  Abschluss des Ticketkaufs erforderlich sind."</PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Für den Abschluss des Ticketkaufs sind folgende Arbeitsschritte
                  erforderlich:
                  1. Dem Kaufer werden die Daten zu den ausgewählten Reisen zur Über-
                     prüfung angezeigt
                  2. Der Kaufer bestätigt diese und initiiert somit eine verbindliche
                     Bestellung"</SLTN_Digest>
   </Solution>
  </Theory>
  <Practice>
```

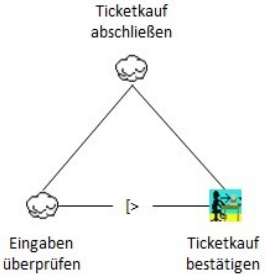
```
  <KnownUses>
   <KnownUse>
    <KNWN_KnownUseID>"0011_0001"</KNWN_KnownUseID>
    <KNWN_Label>"Ticketkauf abschließen - Deutsche Bahn AG"</KNWN_Label>
    <KNWN_Description>"Abschließen des Ticketkaufs im Online-Portal der Deutschen
                       Bahn (www.bahn.de)"</KNWN_Description>
    <KNWN_Images>
     <KNWN_Image>
```


```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0011_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
    <RLTN_PatternName>"Reiseticket kaufen"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0011_0002"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"besteht aus"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0012"</RLTN_PatternID>
    <RLTN_PatternName>"Eingaben überprüfen"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0011_01"</MDFR_FragmentID>
    <MDFR_Label>"Ticketkauf abschließen"</MDFR_Label>
```

<MDFR_Purpose>"Eingaben bestätigen und Ticket anfordern"</MDFR_Purpose>
<MDFR_Annotation>""</MDFR_Annotation>
<MDFR_Diagram>

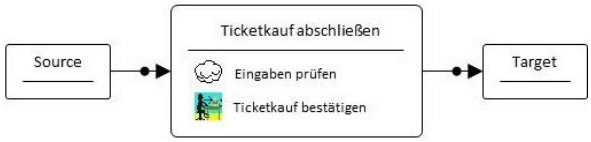


</MDFR_Diagram>
<MDFR_Fragment>
 <TMF_IncludesDummy>False</IncludesDummy>
 <TMF_ProposedTempOp>SequetialEnablingInfo</TMF_ProposedTempOp>
 <TMF_Content>
  <Subtask>
   <TaskID>"__TSK_0011_01_0001"</TaskID>
   <TaskName>"Ticketkauf abschließen"</TaskName>
   <TaskDescription>""</TaskDescription>
   <TaskType>Abstraction</TaskType>
   <TaskOrigin>
    <TaskOriginPatternID>"PPT_0001_0011"</TaskOriginPatternID>
    <TaskOriginPatternName>"Ticketkauf abschließen"</TaskOriginPatternName>
    <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
    <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
   </TaskOrigin>
   <ContextsOfUse></ContextsOfUse>
   <Optional>False</Optional>
   <Iterative>False</Iterative>
   <Conditional>False</Conditional>
   <Preconditions></Preconditions>
   <Postconditions></Postconditions>
   <Position>
    <Parent>
     <ParentID>""</ParentID>
     <ParentName>""</ParentName>
    </Parent>
    <SiblingLeft>
     <SiblingLeftID>""</SiblingLeftID>
     <SiblingLeftName>""</SiblingLeftName>
    </SiblingLeft>
    <SiblingRight>
     <SiblingRightID>""<SiblingRightID>
     <SiblingRightName>""</SiblingRightName>
    </SiblingRight>
   </Position>
   <TemporalOperator>Disabling</TemporalOperator>
   <UIConcepts></UIConcepts>
   <Subtasks>
    <Subtask>
     <TaskID>"__TSK_0011_01_0002"</TaskID>
     <TaskName>"Eingaben überprüfen"</TaskName>
     <TaskDescription>""</TaskDescription>
     <TaskType>Abstraction</TaskType>
     <TaskOrigin>
      <TaskOriginPatternID>"PPT_0001_0011"</TaskOriginPatternID>
      <TaskOriginPatternName>"Ticketkauf abschließen"</TaskOriginPatternName>
      <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
      <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
     </TaskOrigin>
     <ContextsOfUse></ContextsOfUse>
     <Optional>False</Optional>
     <Iterative>False</Iterative>
     <Conditional>False</Conditional>
     <Preconditions></Preconditions>
     <Postconditions></Postconditions>
     <Position>

```
        <Parent>
         <ParentID>"__TSK_0011_01_0001"</ParentID>
         <ParentName>"Ticketkauf abschließen"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>""</SiblingLeftID>
         <SiblingLeftName>""</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>"__TSK_0011_01_0003"<SiblingRightID>
         <SiblingRightName>"Ticketkauf bestätigen"</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>Disabling</TemporalOperator>
       <UIConcepts></UIConcepts>
      </Subtask>
      <Subtask>
       <TaskID>"__TSK_0011_01_0003"</TaskID>
       <TaskName>"Ticketkauf bestätigen"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Interaction</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0011"</TaskOriginPatternID>
        <TaskOriginPatternName>"Ticketkauf abschließen"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>"__TSK_0011_01_0001"</ParentID>
         <ParentName>"Ticketkauf abschließen"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0011_01_0002"</SiblingLeftID>
         <SiblingLeftName>"Eingaben überprüfen"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator></TemporalOperator>
       <UIConcepts>
        <UIConcept>
         <UIConceptID>"__CPT_0011_01_0001"</UIConceptID>
         <UIConceptName>"Ticketkauf bestätigen"</UIConceptName>
        </UIConcept>
       </UIConcepts>
      </Subtask>
     </Subtasks>
    </Subtask>
   </TMF_Content>
  </MDFR_Fragment>
 </ModelFragment>
 <ModelFragment>
  <MDFR_Type>Concept</MDFR_Type>
  <MDFR_FragmentID>"__CMF_0011_01"</MDFR_FragmentID>
  <MDFR_Label>"Konzepte – Ticketkauf bestätigen"</MDFR_Label>
  <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
  </MDFR_Purpose>
  <MDFR_Annotation>""</MDFR_Annotation>
  <MDFR_Diagram>
```

Konzepte des Patterns „Ticketkauf abschließen“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0011_01_0001	Ticketkauf bestätigen	Activator

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <CMF_IncludesDummy>False</CMF_IncludesDummy>
  <CMF_Content>
   <Concept>
    <CCPT_ConceptID>"__CPT_0011_01_0001"</CCPT_ConceptID>
    <CCPT_ConceptName>"Ticketkauf bestätigen"</CCPT_ConceptName>
    <CCPT_Description>"Übertragen der Daten und bestätigen des Ticketkaufs"
    </CCPT_Description>
    <CCPT_Label>"Bestätigen"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>Activator</CCPT_ConceptType>
    <CCPT_DataType>""</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0011"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Ticketkauf abschließen"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0011_01_0003"</CCPT_TaskID>
      <CCPT_TaskName>"Ticketkauf bestätigen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0011_01"</MDFR_FragmentID>
 <MDFR_Label>"Ticketkauf abschließen"</MDFR_Label>
 <MDFR_Purpose>"Eingaben überprüfen und Ticketkauf bestätigen"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```




```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0011_01_0001"</DialogID>
    <DialogName>"Ticketkauf abschließen"</DialogName>
    <DialogDescription>""</DialogDescription>
    <DialogLabel>"Ticketkauf abschließen"</DialogLabel>
```

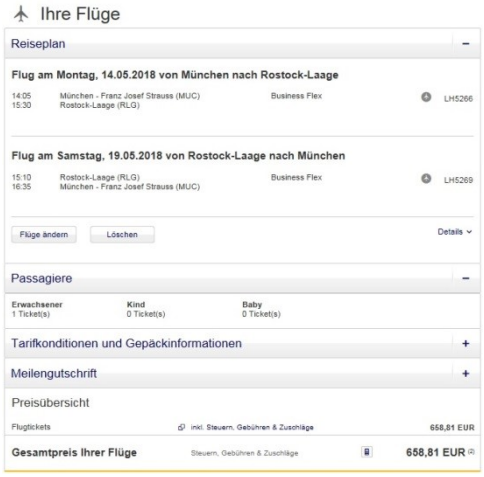
```
        <DialogType>Complex</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>""</PRED_DialogID>
           <PRED_DialogName>""</PRED_DialogName>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>"__CPT_0011_01_0001"</SUCC_ConceptID>
            <SUCC_ConceptName>"Ticketkauf bestätigen"</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <DLG_Tasks>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0011_01_0002"</DLG_TaskID>
          <DLG_TaskName>"Eingaben überprüfen"</DLG_TaskName>
          <DLG_Processing>Recursive</DLG_Processing>
         </DLG_Task>
         <DLG_Task>
          <DLG_TaskID>"__TSK_0011_01_0003"</DLG_TaskID>
          <DLG_TaskName>"Ticketkauf bestätigen"</DLG_TaskName>
          <DLG_Processing>Exclusive</DLG_Processing>
         </DLG_Task>
        </DLG_Tasks>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

J.12 Eingaben überprüfen

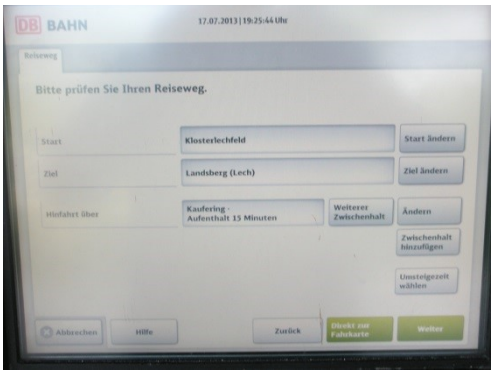
```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0012"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Eingaben überprüfen"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>""</ANCS_PatternCompilationName>
    <ANCS_PatternName>""</ANCS_PatternName>
   </Ancestry>
   <CreationDate>"30.12.2016"</CreationDate>
   <LastModified>"14.05.2018"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0012_0001"</CHNG_ChangeID>
     <CHNG_Date>"14.05.2018"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Der Käufer soll die relevanten Angaben und ausgewählten Reisen vor
                  der eigentlichen Bestellung der Tickets noch einmal überprüfen."
    </PRBL_Digest>
   </Problem>
   <Context>
    <CNTX_Digest>"Es handelt sich um den Kauf von Reisetickets zur Personenbeförderung
                  mit den üblichen Verkehrsmitteln wie beispielsweise Bahn, Bus oder
                  Flugzeug."</CNTX_Digest>
   </Context>
   <Solution>
    <SLTN_Digest>"Es müssen dem Käufer zumindest die wichtigsten Daten hinsichtlich
                  ausgewählten Reiseverbindungen zur Überprüfung vorgelegt werden.
                  Dies sind:
                  1. Die gewählte Hinreise-Verbindung
                  2. Optional die gewählte Rückreise-Verbindung
                  3. Preisangaben
                  4. Weitere Angaben, wie beispielsweise die gewählte Zahlungsart oder
                     Versandart"</SLTN_Digest>
```

```
   </Solution>
  </Theory>
  <Practice>
   <KnownUses>
    <KnownUse>
     <KNWN_KnownUseID>"0012_0001"</KNWN_KnownUseID>
     <KNWN_Label>"Überprüfung der ausgewählten Reisen – Deutsche Bahn AG"</KNWN_Label>
     <KNWN_Description>"Überprüfung der ausgewählten Reiseverbindungen durch den
                        Käufer im Online-Portal der Deutschen Bahn (www.bahn.de)"
     </KNWN_Description>
     <KNWN_Images>
      <KNWN_Image>
```

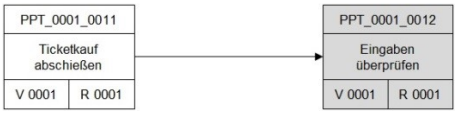

```
      </KNWN_Image>
     </KNWN_Images>
    </KnownUse>
    <KnownUse>
     <KNWN_KnownUseID>"0012_0002"</KNWN_KnownUseID>
     <KNWN_Label>"Überprüfung der ausgewählten Reisen – Deutsche Lufthansa
AG"</KNWN_Label>
     <KNWN_Description>"Überprüfung der ausgewählten Reiseverbindungen durch den
Käufer im
                        Online-Portal der Lufthansa
(www.lufthansa.de)"</KNWN_Description>
     <KNWN_Images>
      <KNWN_Image>
```



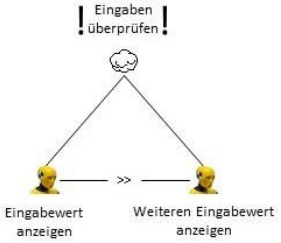

```
      </KNWN_Image>
     </KNWN_Images>
    </KnownUse>
    <KnownUse>
     <KNWN_KnownUseID>"0012_0003"</KNWN_KnownUseID>
     <KNWN_Label>"Überprüfung der ausgewählten Reisen – DB-Fahrkartenautomat
"</KNWN_Label>
     <KNWN_Description>"Überprüfung der ausgewählten Reiseverbindungen durch den
Käufer an
                        einem Fahrkartenautomat der Deutschen Bahn"</KNWN_Description>
     <KNWN_Images>
      <KNWN_Image>
```




```
     </KNWN_Image>
    </KNWN_Images>
   </KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```




```
 </RLTN_Diagram>
 <Relations>
  <Relation>
   <RLTN_RelationID>"RLN_0001_0012_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Komposition</RLTN_Type>
   <RLTN_Sense>"ist Teil von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0011"</RLTN_PatternID>
    <RLTN_PatternName>"Ticketkauf abschließen"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Task</MDFR_Type>
    <MDFR_FragmentID>"__TMF_0012_01"</MDFR_FragmentID>
    <MDFR_Label>"Eingaben überprüfen"</MDFR_Label>
    <MDFR_Purpose>"Zusammenfassung der Eingaben zur Überprüfung durch den Benutzer"
    </MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```




```
    </MDFR_Diagram>
    <MDFR_Fragment>
```

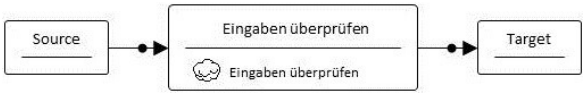
```
<TMF_IncludesDummy>True</IncludesDummy>
<TMF_ProposedTempOp>SequentialEnabling</TMF_ProposedTempOp>
<TMF_Content>
 <Subtask>
  <TaskID>"__TSK_0012_01_0001"</TaskID>
  <TaskName>"Eingaben überprüfen"</TaskName>
  <TaskDescription>""</TaskDescription>
  <TaskType>Abstraction</TaskType>
  <TaskOrigin>
   <TaskOriginPatternID>"PPT_0001_0012"</TaskOriginPatternID>
   <TaskOriginPatternName>"Eingaben überprüfen"</TaskOriginPatternName>
   <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
   <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
  </TaskOrigin>
  <ContextsOfUse></ContextsOfUse>
  <Optional>False</Optional>
  <Iterative>False</Iterative>
  <Conditional>False</Conditional>
  <Preconditions></Preconditions>
  <Postconditions></Postconditions>
  <Position>
   <Parent>
    <ParentID>""</ParentID>
    <ParentName>""</ParentName>
   </Parent>
   <SiblingLeft>
    <SiblingLeftID>""</SiblingLeftID>
    <SiblingLeftName>""</SiblingLeftName>
   </SiblingLeft>
   <SiblingRight>
    <SiblingRightID>""<SiblingRightID>
    <SiblingRightName>""</SiblingRightName>
   </SiblingRight>
  </Position>
  <TemporalOperator>Disabling</TemporalOperator>
  <UIConcepts></UIConcepts>
  <Subtasks>
   <Subtask>
    <TaskID>"__TSK_0012_01_0002"</TaskID>
    <TaskName>"Eingabewert anzeigen"</TaskName>
    <TaskDescription>""</TaskDescription>
    <TaskType>Dummy</TaskType>
    <TaskOrigin>
     <TaskOriginPatternID>"PPT_0001_0012"</TaskOriginPatternID>
     <TaskOriginPatternName>"Eingaben überprüfen"</TaskOriginPatternName>
     <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
     <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
    </TaskOrigin>
    <ContextsOfUse></ContextsOfUse>
    <Optional>False</Optional>
    <Iterative>False</Iterative>
    <Conditional>False</Conditional>
    <Preconditions></Preconditions>
    <Postconditions></Postconditions>
    <Position>
     <Parent>
      <ParentID>"__TSK_0012_01_0001"</ParentID>
      <ParentName>"Eingaben überprüfen"</ParentName>
     </Parent>
     <SiblingLeft>
      <SiblingLeftID>""</SiblingLeftID>
      <SiblingLeftName>""</SiblingLeftName>
     </SiblingLeft>
     <SiblingRight>
      <SiblingRightID>"  TSK_0012_01_0003"<SiblingRightID>
      <SiblingRightName>"Weiteren Eingabewert anzeigen"</SiblingRightName>
     </SiblingRight>
    </Position>
    <TemporalOperator>Sequential Enabling</TemporalOperator>
    <UIConcepts>
     <UIConcept>
```

```
        <UIConceptID>"__CPT_0012_01_00013"</UIConceptID>
        <UIConceptName>"Eingabewert Dummy"</UIConceptName>
       </UIConcept>
      </UIConcepts>
      </Subtask>
      <Subtask>
       <TaskID>"__TSK_0012_01_0003"</TaskID>
       <TaskName>"Weiteren Eingabewert anzeigen"</TaskName>
       <TaskDescription>""</TaskDescription>
       <TaskType>Dummy</TaskType>
       <TaskOrigin>
        <TaskOriginPatternID>"PPT_0001_0012"</TaskOriginPatternID>
        <TaskOriginPatternName>"Eingaben überprüfen"</TaskOriginPatternName>
        <TaskOriginPatternVersion>"0001"</TaskOriginPatternVersion>
        <TaskOriginPatternRevision>"0001"</TaskOriginPatternRevision>
       </TaskOrigin>
       <ContextsOfUse></ContextsOfUse>
       <Optional>False</Optional>
       <Iterative>False</Iterative>
       <Conditional>False</Conditional>
       <Preconditions></Preconditions>
       <Postconditions></Postconditions>
       <Position>
        <Parent>
         <ParentID>"__TSK_0012_01_0001"</ParentID>
         <ParentName>"Eingaben überprüfen"</ParentName>
        </Parent>
        <SiblingLeft>
         <SiblingLeftID>"__TSK_0012_01_0002"</SiblingLeftID>
         <SiblingLeftName>"Eingabewert anzeigen"</SiblingLeftName>
        </SiblingLeft>
        <SiblingRight>
         <SiblingRightID>""<SiblingRightID>
         <SiblingRightName>""</SiblingRightName>
        </SiblingRight>
       </Position>
       <TemporalOperator>Sequential Enabling</TemporalOperator>
       <UIConcepts>
        <UIConcept>
         <UIConceptID>"__CPT_0012_01_0002"</UIConceptID>
         <UIConceptName>"Weiterer Eingabewert Dummy"</UIConceptName>
        </UIConcept>
       </UIConcepts>
      </Subtask>
     </Subtasks>
    </Subtask>
   </TMF_Content>
  </MDFR_Fragment>
 </ModelFragment>
 <ModelFragment>
  <MDFR_Type>Concept</MDFR_Type>
  <MDFR_FragmentID>"__CMF_0012_01"</MDFR_FragmentID>
  <MDFR_Label>"Konzepte – Eingaben prüfen"</MDFR_Label>
  <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
  </MDFR_Purpose>
  <MDFR_Annotation>""</MDFR_Annotation>
  <MDFR_Diagram>
```

Konzepte des Patterns „Eingaben überprüfen“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0012_01_0001	Eingabewert Dummy	DataOutput
2	__CPT_0012_01_0002	Weiterer Eingabewert Dummy	DataOutput

```
  </MDFR_Diagram>
  <MDFR_Fragment>
   <CMF_IncludesDummy>True</CMF_IncludesDummy>
   <CMF_Content>
    <Concept>
     <CCPT_ConceptID>"__CPT_0012_01_0001"</CCPT_ConceptID>
     <CCPT_ConceptName>"Eingabewert Dummy"</CCPT_ConceptName>
     <CCPT_Description>"Eingabewert für die Anzeige"</CCPT_Description>
     <CCPT_Label>"Eingabewert Dummy:"</CCPT_Label>
     <CCPT_Perceptible>True</CCPT_Perceptible>
```

```
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>Dummy</CCPT_ConceptType>
    <CCPT_DataType>"Integer"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0012"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Eingaben überprüfen"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0012_01_0002"</CCPT_TaskID>
      <CCPT_TaskName>"Eingabewert anzeigen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
   <Concept>
    <CCPT_ConceptID>"__CPT_0012_01_0002"</CCPT_ConceptID>
    <CCPT_ConceptName>"Weiterer Eingabewert Dummy"</CCPT_ConceptName>
    <CCPT_Description>"Weiterer Eingabewert für die Anzeige"</CCPT_Description>
    <CCPT_Label>"Weiterer Eingabewert Dummy:"</CCPT_Label>
    <CCPT_Perceptible>True</CCPT_Perceptible>
    <CCPT_Enabled>True</CCPT_Enabled>
    <CCPT_Required>True</CCPT_Required>
    <CCPT_ConceptType>Dummy</CCPT_ConceptType>
    <CCPT_DataType>"Integer"</CCPT_DataType>
    <CCPT_Value>""</CCPT_Value>
    <CCPT_ConceptOrigin>
     <CCPT_OriginPatternID>"PPT_0001_0012"</CCPT_OriginPatternID>
     <CCPT_OriginPatternName>"Eingaben überprüfen"</CCPT_OriginPatternName>
     <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
     <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
    </CCPT_ConceptOrigin>
    <CCPT_Preconditions></CCPT_Preconditions>
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks>
     <CCPT_TaskLink>
      <CCPT_TaskID>"__TSK_0012_01_0003"</CCPT_TaskID>
      <CCPT_TaskName>"Weiteren Eingabewert anzeigen"</CCPT_TaskName>
     </CCPT_TaskLink>
    </CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
</ModelFragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0012_01"</MDFR_FragmentID>
 <MDFR_Label>"Eingaben überprüfen"</MDFR_Label>
 <MDFR_Purpose>"Eingaben überprüfen"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```



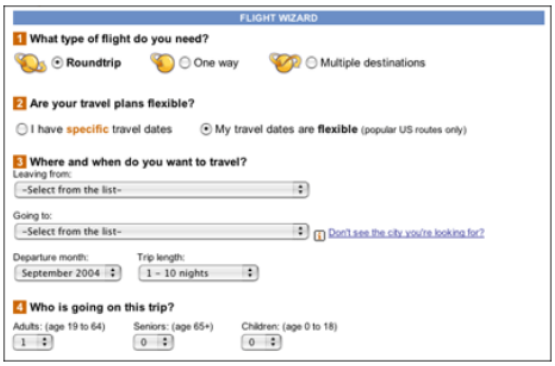

```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>False</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
```

```
      </DMF_ContextModelReference>
      <DMF_ContextModelReference>
       <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
       <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
       <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
       <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
      </DMF_ContextModelReference>
     </DMF_ContextModelReferences>
     <DMF_Content>
      <Dialog>
       <DialogID>"__DLG_0012_01_0001"</DialogID>
       <DialogName>"Eingaben überprüfen"</DialogName>
       <DialogDescription>""</DialogDescription>
       <DialogLabel>"Ticketdaten überprüfen"</DialogLabel>
       <DialogType>Single</DialogType>
       <Position>
        <Predecessors>
         <Predecessor>
          <PRED_DialogID>""</PRED_DialogID>
          <PRED_DialogName>""</PRED_DialogName>
         </Predecessor>
        </Predecessors>
        <Successors>
         <Successor>
          <SUCC_DialogID>""</SUCC_DialogID>
          <SUCC_DialogName>""</SUCC_DialogName>
          <SUCC_TransitionType>Sequential</SUCC_TransitionType>
          <SUCC_Trigger>
           <SUCC_ConceptID>""</SUCC_ConceptID>
           <SUCC_ConceptName>""</SUCC_ConceptName>
          </SUCC_Trigger>
         </Successor>
        </Successors>
       </Position>
       <DLG_Tasks>
        <DLG_Task>
         <DLG_TaskID>"__TSK_0012_01_0001"</DLG_TaskID>
         <DLG_TaskName>"Eingaben überprüfen"</DLG_TaskName>
         <DLG_Processing>Recursive</DLG_Processing>
        </DLG_Task>
       </DLG_Tasks>
      </Dialog>
     </DMF_Content>
    </MDFR_Fragment>
   </ModelFragment>
  </ModelFragments>
 </PaMGIS>
</Deployment>
</Pattern>
```

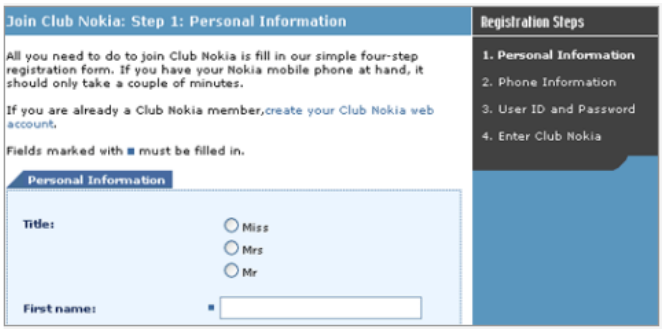
J.13 Wizard

```
<Pattern>
 <Head>
  <Texture>"PPSL 3.0"</Texture>
  <UniquePatternID>
   <UPID_PatternCompilationID>"PPL_0001"</UPID_PatternCompilationID>
   <UPID_PatternID>"PPT_0001_0013"</UPID_PatternID>
   <UPID_VersionNumber>"0001"</UPID_VersionNumber>
   <UPID_RevisionNumber>"0001"</UPID_RevisionNumber>
  </UniquePatternID>
  <Classification>
   <CLSS_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
   </CLSS_PatternCompilationName>
   <CLSS_PatternType>"HCI Design Pattern"</CLSS_PatternType>
  </Classification>
  <Names>
   <Name>"Wizard"</Name>
  </Names>
  <Status>Generally released</Status>
  <Contribution>
   <Authors>
    <Author>
     <AuthorID>"ATR_0001"</AuthorID>
     <Author_FirstName>"Jürgen"</Author_FirstName>
     <Author_LastName>"Engel"</Author_LastName>
    <Author>
   </Authors>
  </Contribution>
  <History>
   <Ancestry>
    <ANCS_PatternCompilationName>"Patterns in Interation Design"
    </ANCS_PatternCompilationName>
    <ANCS_PatternName>"Wizard"</ANCS_PatternName>
    <ANCS_Texture>"Welie"</ANCS_Texture>
    <ANCS_PatternID>"Wizard"</ANCS_PatternID>
    <ANCS_Category>"Basic interactions"</ANCS_Category>
    <ANCS_VersionNumber>""</ANCS_VersionNumber>
    <ANCS_RevisionNumber>""</ANCS_RevisionNumber>
    <ANCS_Link>
     <ANCS_LinkDisplayName>"Welie - Wizard"</ANCS_LinkDisplayName>
     <ANCS_URL>http://www.welie.com/patterns/showPattern.php?patternID=wizard
     </ANCS_URL>
    </ANCS_Link>
   </Ancestry>
   <CreationDate>"28.06.2017"</CreationDate>
   <LastModified>"20.08.2018"</LastModified>
   <Changes>
    <Change>
     <CHNG_ChangeID>"CNG_0001_0013_0001"</CHNG_ChangeID>
     <CHNG_Date>"20.08.2018"</CHNG_Date>
     <CHNG_VersionNumber>"0001"</CHNG_VersionNumber>
     <CHNG_RevisionNumber>"0000"</CHNG_RevisionNumber>
     <CHNG_Originator>"ATR_0001"</CHNG_Originator>
     <CHNG_Description>"Anpassungen in der Deployment-Sektion"</CHNG_Description>
     <CHNG_Reason>"Überarbeitung"</CHNG_Reason>
     <CHNG_Link></CHNG_Link>
    </Change>
   </Changes>
  </History>
 </Head>
 <Body>
  <Theory>
   <Problem>
    <PRBL_Digest>"Ein Benutzer möchte ein Ziel erreichen, für das jedoch mehrere
                  Entscheidungen getroffen werden müssen, bevor es vollständig
                  erreicht ist."</PRBL_Digest>
   </Problem>
```

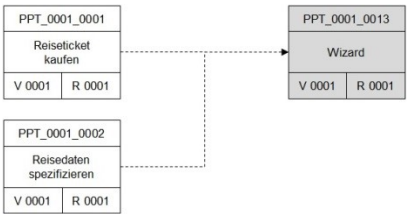
```
 <Context>
  <CNTX_Digest>"Unter folgenden Bedingungen können Probleme bei der Bearbeitung der
               Aufgabe auftreten und somit die Zielerreichung verhindert werden:
               1. Ein unerfahrener Benutzer kann die Komplexität der Aufgabe nicht
                  bewältigen und verliert den Überblick über die benötigten Anga-
                  ben.
               2. Die Abfragen lassen sich auf dem eingesetzten Endgerät nicht alle
                  gleichzeitig darstellen."</CNTX_Digest>
 </Context>
 <Solution>
  <SLTN_Digest>"Die Gesamtaufgabe wird in eine Sequenz von kleineren, logisch zusam-
               men gehörenden Unteraufgaben zerlegt, die nacheinander in separaten
               Einzelschritten abgearbeitet werden können. Optional kann der Benut-
               zer zu bereits durchgeführten Schritten zurückkehren."</SLTN_Digest>
 </Solution>
</Theory>
<Practice>
 <KnownUses>
  <KnownUse>
   <KNWN_KnownUseID>"0013_0001"</KNWN_KnownUseID>
   <KNWN_Label>"Flug-Auskunft von expedia.com"</KNWN_Label>
   <KNWN_Description>"Schrittweise Angabe von Daten zur Suche von Flugverbindungen
                      bei expedia.com"</KNWN_Description>
   <KNWN_Link>
    <KNWN_LinkDisplayName>"Internet-Seite von expedia.com"</KNWN_LinkDisplayName>
    <KNWN_URL>"http://expedia.com"</KNWN_URL>
   </KNWN_link>
   <KNWN_Images>
    <KNWN_Image>
```




```
    </KNWN_Image>
   </KNWN_Images>
   <KnownUse>
   <KnownUse>
    <KNWN_KnownUseID>"0013_0002"</KNWN_KnownUseID>
    <KNWN_Label>"Club-Registrierung bei Nokia"</KNWN_Label>
    <KNWN_Description>"Schrittweise Angabe von Daten zur Registrierung bei einem Club
                       der Firma Nokia"</KNWN_Description>
    <KNWN_Link>
     <KNWN_LinkDisplayName>"Internet-Seite der Firma Nokia"</KNWN_LinkDisplayName>
     <KNWN_URL>"http://www.club.nokia.com/"</KNWN_URL>
    </KNWN_link>
    <KNWN_Images>
     <KNWN_Image>
```



```
    </KNWN_Image>
   </KNWN_Images>
   <KnownUse>
  </KnownUses>
 </Practice>
</Body>
<Relationship>
 <RLTN_Information>""</RLTN_Information>
 <RLTN_Diagram>
```



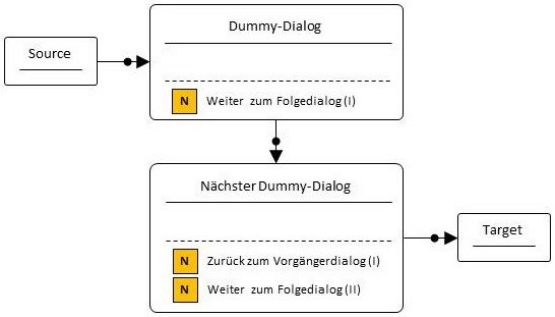

```
 </RLTN_Diagram>
 <Relations>
  </Relation>
   <RLTN_RelationID>"RLN_0001_0013_0001"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Assoziation</RLTN_Type>
   <RLTN_Sense>"wird genutzt von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0001"</RLTN_PatternID>
    <RLTN_PatternName>"Reiseticket kaufen"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
  </Relation>
   <RLTN_RelationID>"RLN_0001_0013_0002"</RLTN_RelationID>
   <RLTN_Nature>Internal</RLTN_Nature>
   <RLTN_Type>Assoziation</RLTN_Type>
   <RLTN_Sense>"wird genutzt von"</RLTN_Sense>
   <RLTN_Reference>
    <RLTN_PatternCompilationID>"PPL_0001"</RLTN_PatternCompilationID>
    <RLTN_PatternCompilationName>"Ticket-Verkauf für Personenbeförderung"
    </RLTN_PatternCompilationName>
    <RLTN_PatternID>"PPT_0001_0002"</RLTN_PatternID>
    <RLTN_PatternName>"Reisedaten spezifizieren"</RLTN_PatternName>
    <RLTN_VersionNumber>"0001"</RLTN_VersionNumber>
    <RLTN_RevisionNumber>"0001"</RLTN_RevisionNumber>
   </RLTN_Reference>
  </Relation>
 </Relations>
</Relationship>
<Deployment>
 <PaMGIS>
  <ModelFragments>
   <ModelFragment>
    <MDFR_Type>Concept</MDFR_Type>
    <MDFR_FragmentID>"__CMF_0013_01"</MDFR_FragmentID>
    <MDFR_Label>"Konzepte – Wizard"</MDFR_Label>
    <MDFR_Purpose>"Spezifikation aller für das Pattern relevanten Konzepte"
    </MDFR_Purpose>
    <MDFR_Annotation>""</MDFR_Annotation>
    <MDFR_Diagram>
```

Konzepte des Patterns „Wizard“			
Nr.	**Konzept-ID**	**Konzept-Name**	**Konzept-Typ**
1	__CPT_0013_01_0001	Weiter zum Folgedialog Dummy (I)	Navigator
2	__CPT_0013_01_0002	Zurück zum Vorgängerdialog Dummy (I)	Navigator
3	__CPT_0013_01_0002	Weiter zum Folgedialog Dummy (II)	Navigator

```
    </MDFR_Diagram>
    <MDFR_Fragment>
     <CMF_IncludesDummy>True</CMF_IncludesDummy>
     <CMF_Content>
```

```
<Concept>
 <CCPT_ConceptID>"__CPT_0013_01_0001"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weiter zum Folgedialog Dummy (I)"</CCPT_ConceptName>
 <CCPT_Description>"Zusätzliches Konzept, das bei der Dialog-
                   modellierung im Dialog-Modell-Fragment benötigt wird"
 </CCPT_Description>
 <CCPT_Label>"Weiter"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0013"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Wizard"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks></CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0013_01_0002"</CCPT_ConceptID>
 <CCPT_ConceptName>"Zurück zum Vorgängerdialog Dummy (I)"</CCPT_ConceptName>
 <CCPT_Description>"Zusätzliches Konzept, das bei der Dialog-
                   modellierung im Dialog-Modell-Fragment benötigt wird"
 </CCPT_Description>
 <CCPT_Label>"Zurück"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0013"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Wizard"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
 <CCPT_Postconditions><CCPT_Postconditions>
 <CCPT_DataLink></CCPT_DataLink>
 <CCPT_TaskLinks></CCPT_TaskLinks>
</Concept>
<Concept>
 <CCPT_ConceptID>"__CPT_0013_01_0003"</CCPT_ConceptID>
 <CCPT_ConceptName>"Weiter zum Folgedialog Dummy (II)"</CCPT_ConceptName>
 <CCPT_Description>"Zusätzliches Konzept, das bei der Dialog-
                   modellierung im Dialog-Modell-Fragment benötigt wird"
 </CCPT_Description>
 <CCPT_Label>"Weiter"</CCPT_Label>
 <CCPT_Perceptible>True</CCPT_Perceptible>
 <CCPT_Enabled>True</CCPT_Enabled>
 <CCPT_Required>True</CCPT_Required>
 <CCPT_ConceptType>Dummy</CCPT_ConceptType>
 <CCPT_DataType>""</CCPT_DataType>
 <CCPT_Value>""</CCPT_Value>
 <CCPT_ConceptOrigin>
  <CCPT_OriginPatternID>"PPT_0001_0013"</CCPT_OriginPatternID>
  <CCPT_OriginPatternName>"Wizard"
  </CCPT_OriginPatternName>
  <CCPT_OriginPatternVersion>"0001"</CCPT_OriginPatternVersion>
  <CCPT_OriginPatternRevision>"0001"</CCPT_OriginPatternRevision>
 </CCPT_ConceptOrigin>
 <CCPT_Preconditions></CCPT_Preconditions>
```

```
    <CCPT_Postconditions><CCPT_Postconditions>
    <CCPT_DataLink></CCPT_DataLink>
    <CCPT_TaskLinks></CCPT_TaskLinks>
   </Concept>
  </CMF_Content>
 </MDFR_Fragment>
<ModelFragment>
 <MDFR_Type>Dialog</MDFR_Type>
 <MDFR_FragmentID>"__DMF_0013_01"</MDFR_FragmentID>
 <MDFR_Label>"Wizard"</MDFR_Label>
 <MDFR_Purpose>"Serie von Dialogen mit Konzepten zur Navigation zum jeweils
                nächsten und vorhergehenden Dialog"</MDFR_Purpose>
 <MDFR_Annotation>"Anwendbar in allen Kontexten der Fallstudie"</MDFR_Annotation>
 <MDFR_Diagram>
```




```
 </MDFR_Diagram>
 <MDFR_Fragment>
  <DMF_IncludesDummy>True</DMF_IncludesDummy>
  <DMF_ContextModelReferences>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0001"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Großer Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
   <DMF_ContextModelReference>
    <DMF_ContextModelID>"CTX_0002"</DMF_ContextModelID>
    <DMF_ContextModelVersion>"0001"</DMF_ContextModelVersion>
    <DMF_ContextModelRevision>"0000"</DMF_ContextModelRevision>
    <DMF_ContextModelName>"Kleiner Bildschirm"</DMF_ContextModelName>
   </DMF_ContextModelReference>
  </DMF_ContextModelReferences>
  <DMF_Content>
   <Dialog>
    <DialogID>"__DLG_0013_01_0001"</DialogID>
    <DialogName>"Dummy-Dialog"</DialogName>
    <DialogDescription>"Erster Dialog in der Sequenz"</DialogDescription>
    <DialogLabel>""</DialogLabel>
    <DialogType>Dummy</DialogType>
    <Position>
     <Predecessors>
      <Predecessor>
       <PRED DialogID>""</PRED DialogID>
       <PRED_DialogName>""</PRED_DialogName>
      </Predecessor>
     </Predecessors>
     <Successors>
      <Successor>
       <SUCC DialogID>"  DLG 0013 01 0002"</SUCC DialogID>
       <SUCC DialogName>"Nächster Dummy-Dialog"</SUCC DialogName>
       <SUCC_TransitionType>Sequential</SUCC_TransitionType>
       <SUCC_Trigger>
        <SUCC_ConceptID>"__CPT_0013_01_0001"</SUCC_ConceptID>
        <SUCC_ConceptName>"Weiter zum Folgedialog Dummy(I)"</SUCC_ConceptName>
       </SUCC_Trigger>
      </Successor>
     </Successors>
    </Position>
    <DLG_Tasks></DLG_Tasks>
```

```
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"  CPT 0013 01 0001"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog Dummy (I)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
       <Dialog>
        <DialogID>"  DLG 0013 01 0002"</DialogID>
        <DialogName>"Nächster Dummy-Dialog"</DialogName>
        <DialogDescription>"Folgedialog in der Sequenz"</DialogDescription>
        <DialogLabel>""</DialogLabel>
        <DialogType>Dummy</DialogType>
        <Position>
         <Predecessors>
          <Predecessor>
           <PRED_DialogID>"__DLG_0013_01_0001"</PRED_DialogID>
           <PRED_DialogName>"Dummy-Dialog"</PRED_DialogName>
           <PRED_Trigger>
            <PRED_ConceptID>"  CPT 0013 01 0002"</PRED_ConceptID>
            <PRED_ConceptName>"Zurück zum Vorgängerdialog Dummy (I)"
            </PRED_ConceptName>
           </PRED_Trigger>
          </Predecessor>
         </Predecessors>
         <Successors>
          <Successor>
           <SUCC_DialogID>""</SUCC_DialogID>
           <SUCC_DialogName>""</SUCC_DialogName>
           <SUCC_TransitionType>Sequential</SUCC_TransitionType>
           <SUCC_Trigger>
            <SUCC_ConceptID>"__CPT_0013_01_0003"</SUCC_ConceptID>
            <SUCC_ConceptName>"Weiter zum Folgedialog Dummy (II)"</SUCC_ConceptName>
           </SUCC_Trigger>
          </Successor>
         </Successors>
        </Position>
        <SupplementaryConcepts>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0013_01_0002"</SupplementaryConceptID>
          <SupplementaryConceptName>"Zurück zum Vorgängerdialog Dummy (I)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
         <SupplementaryConcept>
          <SupplementaryConceptID>"__CPT_0013_01_0003"</SupplementaryConceptID>
          <SupplementaryConceptName>"Weiter zum Folgedialog Dummy (II)"
          </SupplementaryConceptName>
         </SupplementaryConcept>
        </SupplementaryConcepts>
       </Dialog>
      </DMF_Content>
     </MDFR_Fragment>
    </ModelFragment>
   </ModelFragments>
  </PaMGIS>
 </Deployment>
</Pattern>
```

K. PaMGIS-Modelle zur Fallstudie

In den folgenden Unterkapiteln befinden sich die Spezifikationen der in der Fallstudie zum Einsatz kommenden Modelle.

Aus Platz- und Übersichtsgründen sind im Wesentlichen jeweils nur die verpflichtenden und die für das Verständnis der Fallstudie benötigten Beschreibungselemente angegeben.

Des Weiteren sind bei allen Zeilen, die über die Seitenbreite hinausgehen, entsprechende Zeilenumbrüche eingefügt und nachfolgende Texteinrückungen vorgenommen worden. Diese gehören zwar nicht zu den eigentlichen Inhalten, erhöhen aber die Lesbarkeit der Spezifikationen im ausgedruckten Format erheblich.

K.1 Kontext-Modell (Großer Bildschirm)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>"CTX_0001"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Context Model</ModelType>
 <ModelName>"Großer Bildschirm"</ModelName>
 <ModelDescription>"Benutzer: Standard-Benutzer,
                    Gerät:    Standard-Desktop-PC,
                    Toolkit:  Python/Tkinter,
                    Umgebung: Standard-Büroumgebung"
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <ContextModelReferences>
   <UserModelReference>
    <USRREF_ModelID>"USR_0001"<USRREF_ModelID>
    <USRREF_ModelVersion>"0001"</USRREF_ModelVersion>
    <USRREF_ModelRevision>"0000"</USRREF_ModelRevision>
    <USRREF_ModelName>"Standard-Benutzer"</USRREF_ModelName>
   <UserModelReference>
   <DeviceModelReference>
    <DEVREF_ModelID>"DEV_0001"<DEVREF_ModelID>
    <DEVREF_ModelVersion>"0001"</DEVREF_ModelVersion>
    <DEVREF_ModelRevision>"0000"</DEVREF_ModelRevision>
    <DEVREF_ModelName>"Standard-Desktop-PC"</DEVREF_ModelName>
   <DeviceModelReference>
   <ToolkitModelReference>
    <TLKREF_ModelID>"TLK_0001"<TLKREF_ModelID>
    <TLKREF_ModelVersion>"0001"</TLKREF_ModelVersion>
    <TLKREF_ModelRevision>"0000"</TLKREF_ModelRevision>
    <TLKREF_ModelName>"Python/Tkinter"</TLKREF_ModelName>
   <ToolkitModelReference>
   <EbnvironmentModelReference>
    <ENVREF_ModelID>"ENV_0001"<ENVREF_ModelID>
    <ENVREF_ModelVersion>"0001"</ENVREF_ModelVersion>
    <ENVREF_ModelRevision>"0000"</ENVREF_ModelRevision>
    <ENVREF_ModelName>"Standard-Büroumgebung"</ENVREF_ModelName>
   <EnvironmentModelReference>
  </ContextModelReferences>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

K.2 Kontext-Modell (Kleiner Bildschirm)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>"CTX_0002"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Context Model</ModelType>
 <ModelName>"Kleiner Bildschirm"</ModelName>
 <ModelDescription>"Benutzer: Standard-Benutzer,
                    Gerät:    Standard-Smart-Phone,
                    Toolkit:  Python/Tkinter,
                    Umgebung: Standard-Büroumgebung"
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <ContextModelReferences>
   <UserModelReference>
    <USRREF_ModelID>"USR_0001"<USRREF_ModelID>
    <USRREF_ModelVersion>"0001"</USRREF_ModelVersion>
    <USRREF_ModelRevision>"0000"</USRREF_ModelRevision>
    <USRREF_ModelName>"Standard-Benutzer"</USRREF_ModelName>
   <UserModelReference>
   <DeviceModelReference>
    <DEVREF_ModelID>"DEV_0002"<DEVREF_ModelID>
    <DEVREF_ModelVersion>"0001"</DEVREF_ModelVersion>
    <DEVREF_ModelRevision>"0000"</DEVREF_ModelRevision>
    <DEVREF_ModelName>"Standard-Smart-Phone"</DEVREF_ModelName>
   <DeviceModelReference>
   <ToolkitModelReference>
    <TLKREF_ModelID>"TLK_0001"<TLKREF_ModelID>
    <TLKREF_ModelVersion>"0001"</TLKREF_ModelVersion>
    <TLKREF_ModelRevision>"0000"</TLKREF_ModelRevision>
    <TLKREF_ModelName>"Python/Tkinter"</TLKREF_ModelName>
   <ToolkitModelReference>
   <EbnvironmentModelReference>
    <ENVREF_ModelID>"ENV_0001"<ENVREF_ModelID>
    <ENVREF_ModelVersion>"0001"</ENVREF_ModelVersion>
    <ENVREF_ModelRevision>"0000"</ENVREF_ModelRevision>
    <ENVREF_ModelName>"Standard-Büroumgebung"</ENVREF_ModelName>
   <EnvironmentModelReference>
  </CoUModelReferences>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

K.3 Benutzer-Modell (Standard-Benutzer)

```
<PAMGIS_Model>
 <UniqueModelID>
  <ModelID>"USR_0001"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>User Model</ModelType>
 <ModelName>"Standard-Benutzer"</ModelName>
 <ModelDescription>"Benutzer ohne signifikante körperliche und geistige
                    Einschränkungen, mit fortgeschrittenen Kenntnissen hinsichtlich
                    Anwendungsdomäne und Systembedienung, wenig Ablenkung bei der
                    Systembedienung und voller Geschäftsfähigkeit."
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <UserCharacteristics>
```

```
    <UserAbilities>
     <USRUA_Visual>High</USRUA_Visual>
     <USRUA_Acoustic>High</USRUA_Acoustic>
     <USRUA_Motor>High</USRUA_Motor>
     <USRUA_Mental>High</USRUA_Mental>
    </UserAbilities>
    <UserExperiences>
     <USRUE_Domain>Advanced</USRUE_Domain>
     <USRUE_Handling>Advanced</USRUE_Handling>
    </UserExperiences>
    <UserDistraction>Low</UserDistraction>
    <UserLegalCapacity>Full</UserLegalCapacity>
    <UserEmotionalState>""</UserEmotionalState>
   </UserCharacteristics>
  </ModelSubjectMatter>
</PAMGIS_Model>
```

K.4 Geräte-Modell (Standard-Desktop-PC)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>"DEV_0001"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Device Model</ModelType>
 <ModelName>"Standard-Desktop-PC"</ModelName>
 <ModelDescription>"Desktop-PC mit Tastatur, Maus und 24-Zoll-Monitor"
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <DeviceCharacteristics>
   <DeviceOperatingSystems>
    <DeviceOperatingSystem>"Windows 7"</DeviceOperatingSystem>
   </DeviceOperatingSystems>
   <DeviceInputFacilities>
    <DEVIF_Motor>
     <DEVIF_Keyboard>Hardware</DEVIF_Keyboard>
     <DEVIF_TouchScreen>No<DEVIF_TouchScreen>
     <DEVIF_PointingDevice>Yes</DEVIF_PointingDevice>
    </DEVIF_Motor>
    <DEVIF_Acoustic>
     <DEVIF_VoiceControl>No</DEVIF_VoiceControl>
    </DEVIF_Acoustic>
    <DEVIF_Visual>
     <DEVIF_GestureControl>No</DEVIF_GestureControl>
     <DEVIF_EyeTracking>No</DEVIF_EyeTracking>
    </DEVIF_Visual>
   </DeviceInputFacilities>
   <DeviceOutputFacilities>
    <DEVOF_Visual>
     <DEVOF_Screen>
      <DEVOF_ScreenSize>"24 Zoll"</DEVOF_ScreenSize>
      <DEVOF_ScreenResolution>"1920x1080"</DEVOF_ScreenResolution>
     </DEVOF_Screen>
    </DEVOF_Visual>
    <DEVOF_Acoustic>
     <DEVOF_Speaker>Yes</DEVOF_Speaker>
     <DEVOF_Headphones>No</DEVOF_Headphones>
    </DEVOF_Acoustic>
    <DEVOF_Tactile>
     <DEVOF_Vibration>No</DEVOF_Vibration>
     <DEVOF_Braille>No</DEVOF_Braille>
    </DEVOF_Tactile>
   </DeviceOutputFacilities>
  </DeviceCharacteristics>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

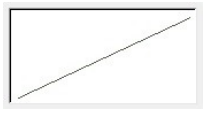
K.5 Geräte-Modell (Standard-Smart-Phone)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>"DEV_0002"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Device Model</ModelType>
 <ModelName>"Standard-Smart-Phone"</ModelName>
 <ModelDescription>"Smart-Phone mit Touch-Screen, Gestensteuerung, Vibrationsalarm und
                    4-Zoll-Bildschirm"
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <DeviceCharacteristics>
   <DeviceOperatingSystems>
    <DeviceOperatingSystem>"Windows Phone 8.1"</DeviceOperatingSystem>
   </DeviceOperatingSystems>
   <DeviceInputFacilities>
    <DEVIF_Motor>
     <DEVIF_Keyboard>Software</DEVIF_Keyboard>
     <DEVIF_TouchScreen>Yes<DEVIF_TouchScreen>
     <DEVIF_PointingDevice>No</DEVIF_PointingDevice>
    </DEVIF_Motor>
    <DEVIF_Acoustic>
     <DEVIF_VoiceControl>No</DEVIF_VoiceControl>
    </DEVIF_Acoustic>
    <DEVIF_Visual>
     <DEVIF_GestureControl>No</DEVIF_GestureControl>
     <DEVIF_EyeTracking>No</DEVIF_EyeTracking>
    </DEVIF_Visual>
   </DeviceInputFacilities>
   <DeviceOutputFacilities>
    <DEVOF_Visual>
     <DEVOF_Screen>
      <DEVOF_ScreenSize>"5 Zoll"</DEVOF_ScreenSize>
      <DEVOF_ScreenResolution>"540x690"</DEVOF_ScreenResolution>
     </DEVOF_Screen>
    </DEVOF_Visual>
    <DEVOF_Acoustic>
     <DEVOF_Speaker>Yes</DEVOF_Speaker>
     <DEVOF_Headphones>No</DEVOF_Headphones>
    </DEVOF_Acoustic>
    <DEVOF_Tactile>
     <DEVOF_Vibration>Yes</DEVOF_Vibration>
     <DEVOF_Braille>No</DEVOF_Braille>
    </DEVOF_Tactile>
   </DeviceOutputFacilities>
  </DeviceCharacteristics>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

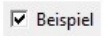
K.6 Toolkit-Modell (Python/Tkinter)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>">TLK_0001"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Toolkit Model</ModelType>
 <ModelName>"Python/Tkinter"</ModelName>
 <ModelDescription>"Toolkit-Modell für Tk unter Python (Tkinter)"
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences>
  <ModelReference>
   <MOREF_LinkDisplayName>"Tkinter – Python interface to Tcl/TK"
   </MOREF_LinkDisplayName>
   <MOREF_URL>"https://docs.python.org/2/library/Tkinter.html"</MOREF_URL>
  </Model_Reference>
 </ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 <ModelSubjectMatter>
  <ToolkitCharacteristics>
   <SupportedOSPlatforms>
   <SupportedOSPlatform>"Windows"</SupportedOSPlatform>
   <SupportedOSPlatform>"Linux"</SupportedOSPlatform>
   <SupportedOSPlatform>"Mac OS X"</SupportedOSPlatform>
   <SupportedOSPlatform>"AS/400"</SupportedOSPlatform>
   <SupportedOSPlatform>"Solaris"</SupportedOSPlatform>
   <SupportedOSPlatform>"HP-UX"</SupportedOSPlatform>
   <SupportedOSPlatform>"OS/390"</SupportedOSPlatform>
   </SupportedOSPlatforms>
   <SupportedUIObjects>
    <SupportedUIObject>
    <TLKSO_Name>"Button"</TLKSO_Name>
    <TLKSO_Description>"Ein Button ist ein Schaltknopf, mit dem Ereignisse ausgelöst
                        werden können, z. B. für Funktions- oder Dialogaufrufe."
    </TLKSO_Description>
    <TLKSO_Diagram>
```


```
    </TLKSO_Diagram>
   </SupportedUIObject>
   <SupportedUIObject>
    <TLKSO_Name>"Canvas"</TLKSO_Name>
    <TLKSO_Description>"Canvas bezeichnet einen rechteckigen Bildschirmbereich zum
                        Erstellen von Zeichnungen oder anderweitiger komplexer
                        Layouts. Es erlaubt die Positionierung von Grafikelementen
                        sowie Text-, Frame- oder anderer UI-Elementen."
    </TLKSO_Description>
    <TLKSO_Diagram>
```



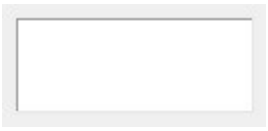
```
    </TLKSO_Diagram>
   </SupportedUIObject>
   <SupportedUIObject>
    <TLKSO_Name>"Checkbutton"</TLKSO_Name>
    <TLKSO_Description>"Ein Checkbutton (oft auch als Checkbox bezeichnet) ermöglichen
                        eine binäre Selektion, z. B. zur Aus- oder Abwahl einer
                        Option."
    </TLKSO_Description>
    <TLKSO_Diagram>
```




```
    </TLKSO_Diagram>
   </SupportedUIObject>
   <SupportedUIObject>
```

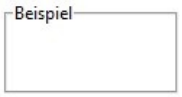
```
 <TLKSO_Name>"Entry"</TLKSO_Name>
 <TLKSO_Description>"Entry dient zur Ein- und Ausgabe von einzeiligem Text
                    (String). Für andere Datentypen, wie beispielsweise Integer
                    oder Float müssen entsprechende Typumwandlungen stattfinden."
 </TLKSO_Description>
 <TLKSO_Diagram>
```


```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Frame"</TLKSO_Name>
 <TLKSO_Description>"Ein Frame ist ein rechteckiger Bereich, der als Container für
                    andere UI-Elemente fungiert."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

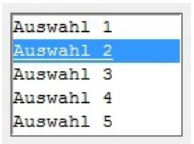


```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Label"</TLKSO_Name>
 <TLKSO_Description>"Ein Label dient zur Anzeige von ein- oder mehrzeiligem Text."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

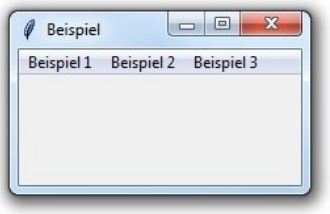

```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"LabelFrame"</TLKSO_Name>
 <TLKSO_Description>"Ein LabelFrame erfüllt im Wesentlichen die gleichen Aufgaben
                    wie ein Frame-Element. Im Unterschied zu diesem kann er aber
                    ein in den Rahmen integriertes Label tragen."
 </TLKSO_Description>
 <TLKSO_Diagram>
```




```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Listbox"</TLKSO_Name>
 <TLKSO_Description>"Mit einer Listbox werden mehrere Textzeilen gleicher
                    Formatierung als Liste dargestellt. Aus dieser können eine
                    oder mehrere Zeilen ausgewählt werden."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

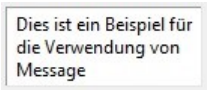



```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
<TLKSO_Name>"Menu"</TLKSO_Name>
 <TLKSO_Description>"Ein Menu ist ein UI-Element zur Auswahl von Optionen. Diese
                    werden direkt unter der Titelzeile eines Fensters
                    dargestellt."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

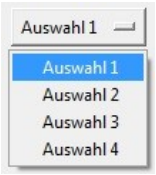



```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Menubutton"</TLKSO_Name>
 <TLKSO_Description>"Ein Menubutton ist der Teil eines Drop-Down-Menüs, der stets
                     sichtbar ist. Wenn er angeklickt wird, dann öffnet sich ein
                     Fenster, das die auswählbaren Optionen anzeigt."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

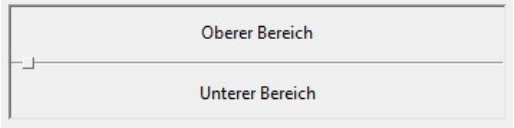

```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Message"</TLKSO_Name>
 <TLKSO_Description>"Die Funktion eines Message-Elements ist sehr ähnlich der eines
                     Labels. Es dient zur Anzeige von mehrzeiligem Text gleicher
                     Formatierung."
 </TLKSO_Description>
 <TLKSO_Diagram>
```




```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"OptionMenu"</TLKSO_Name>
 <TLKSO_Description>"Ein OptionMenu erlaubt die Auswahl einer aus einer
                     vorgegebenen Menge von Optionen."
 </TLKSO_Description>
 <TLKSO_Diagram>
```




```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"PanedWindow"</TLKSO_Name>
 <TLKSO_Description>"Ein PanedWindow ist ähnlich wie ein Frame ein Container für
                     UI-Elemente. Es ist aber in verschiedene Bereiche unterteilt,
                     deren Größe durch Ziehen interaktiv verändert werden kann."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

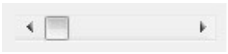



```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Radiobutton"</TLKSO_Name>
```

```
 <TLKSO_Description>"Mithilfe einer Gruppe von Radiobuttons kann eine aus einer
                    Menge von Optionen ausgewählt werden. Ein Radiobutton besteht
                    aus einem Label und einem Auswahlindikator."
 </TLKSO_Description>
 <TLKSO_Diagram>
```


```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Scale"</TLKSO_Name>
 <TLKSO_Description>"Ein Scale-Element erlaubt das Setzen eines Integer- oder
                    Float-Wertes innerhalb eines definierten Bereichs mittels
                    eines Schiebereglers."
 </TLKSO_Description>
 <TLKSO_Diagram>
```

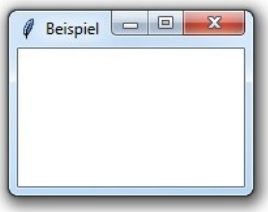

```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Scrollbar"</TLKSO_Name>
 <TLKSO_Description>"Einige UI-Elemente können so konfiguriert sein, dass sie nur
                    einen Ausschnitt ihres eigentlichen Inhalts zeigen (z. B.
                    Listbox oder Canvas). Mit einem Scrollbar kann der sichtbare
                    Bereich vom Benutzer verändert bzw. verschoben werden."
 </TLKSO_Description>
 <TLKSO_Diagram>
```



```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Spinbox"</TLKSO_Name>
 <TLKSO_Description>"Eine Spinbox dient zur Auswahl eines Wertes aus einer
                    vorgegebenen Liste, z. B. aus einem Zahlenbereich oder einer
                    vordefinierten Menge von Zeichenketten. Grafisch besteht sie
                    dieses Element aus einem Bereich für die Anzeige des aktuell
                    ausgewählten Wertes sowie zwei Pfeilen für die Navigation
                    durch den Wertebereich."
 </TLKSO_Description>
 <TLKSO_Diagram>
```


```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Text"</TLKSO_Name>
 <TLKSO_Description>"Ein Text-Element ist ein rechteckiger Bereich mithilfe dessen
                    mehrzeiliger Text angezeigt und bearbeitet werden kann."
 </TLKSO_Description>
 <TLKSO_Diagram>
```


```
 </TLKSO_Diagram>
</SupportedUIObject>
<SupportedUIObject>
 <TLKSO_Name>"Toplevel"</TLKSO_Name>
 <TLKSO_Description>"Das Toplevel-Element ist ein Fenster, das unabhängig von
                    anderen Fenstern beliebig verschoben und in seiner Größe
                    verändert werden kann."
 </TLKSO_Description>
 <TLKSO_Diagram>
```




```
    </TLKSO_Diagram>
   </SupportedUIObject>
   <ToolkitSupportedUIObjects>
  <ToolkitCharacteristics>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

K.7 Umgebungs-Modell (Standard-Büroumgebung)

```
<PaMGIS_Model>
 <UniqueModelID>
  <ModelID>"ENV_0001"</ModelID>
  <ModelVersion>"0001"</ModelVersion>
  <ModelRevision>"0000"</ModelRevision>
 </UniqueModelID>
 <ModelType>Environment Model</ModelType>
 <ModelName>"Standard-Büroumgebung"</ModelName>
 <ModelDescription>"Büroumgebung mit minimaler Beeinträchtigung durch Lichtreflexion,
                    Umgebungsgeräusche und Verschmutzung."
 </ModelDescription>
 <ModelAnnotation>""</ModelAnnotation>
 <ModelReferences></ModelReferences>
 <ModelUserComments></ModelUserComments>
 <ModelUsabilityFeedback>""</ContextModelUsabilityFeedback>
 </ModelSubjectMatter>
  <EnvironmentCharacteristics>
   <LuminousReflectance>Low</LuminousReflectance>
   <AmbientNoise>Low</AmbientNoise>
   <Pollutant>Low</Pollutant>
  </EnvironmentCharacteristics>
 </ModelSubjectMatter>
</PaMGIS_Model>
```

L. Administratives

L.1 Selbständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Jürgen Engel

L.2 Erklärung zur Bewerbung um den Doktorgrad

Hiermit erkläre ich, dass ich mich zuvor weder an der Universität Rostock, noch an einer anderen Universität um den Doktorgrad beworben habe.

Jürgen Engel

L.3 Lebenslauf

Aus Datenschutzgründen wird der Lebenslauf an dieser Stelle nicht veröffentlicht.

L.4 Liste der Veröffentlichungen

2017 Märtin, C., Herdin, C., Engel, J.: Model-based User-Interface Adaptation by Exploiting Situations, Emotions and Software Patterns, Proc. CHIRA 2017, Funchal, Madeira, Portugal, 31 October-2 November, SCITEPRESS (2017)

Engel, J., Märtin, C., Forbrig, P.: Practical Aspects of Pattern-supported Model-driven User Interface Generation. Proc. HCI International 2017, Vancouver, Canada, 9-14 July, Vol. I, Springer LNCS, 2017, pp. 397-414

2016 Engel, J., Märtin, C., Forbrig, P.: A Unified Pattern Specification Formalism to Support User Interface Generation. In Proc. of HCII 2016, Toronto, Canada, 17-22 July, Vol. I, Springer LNCS, 2016, pp. 445-456

2015 Märtin, C., Engel, J., Fesenmeier, A., Reitböck, R., Herdin, C.: Werkzeugunterstützung für die Patternund modellgetriebene Entwicklung interaktiver Systeme in PaMGIS 2.0, Forum Medientechnik – Next Generation, New Ideas, St. Pölten, Austria, 26-27. November 2015, Seidl. M. and Schmiedl, G. (eds.), vwh, 2015, pp. 111-124

Engel, J., Märtin, C., Forbrig, P.: A Concerted Model-driven and Pattern-based Framework for Developing User Interfaces of Interactive Ubiquitous Applications, Proc. Workshop on Large-scale and model-based Interactive Systems: Approaches and Challenges, Duisburg, Germany, June 23, 2015, pp. 35-41

Engel, J., Herdin, C., Märtin, C.: A Review of HCI Pattern Tools, Proc. IHCI 2015, Las Palmas de Gran Canaria, Spain, July 22-24, IADIS Press, 2015, pp. 51-58

2014 Engel, J., Märtin, C., Herdin, C.: Furnishing HCI Patterns to Support Modeling and Generation of Interactive User Interfaces, Proc. 7. Forum Medientechnik, St. Pölten, Austria – Next Generation, New Ideas, vwh, 2014, pp. 27-41

Engel, J., Herdin, C., Märtin, C.: Evaluation of Model-based User Interface Development Approaches, Proc. of HCII 2014, Heraklion, Crete, Grece, 22-27 June, 2014, Springer LNCS, HCI(I), pp. 295-307

Engel, J., Herdin, C., Märtin, C.: Review of User Interface Description Languages, Proc. 6. Forum Medientechnik, St. Pölten, Austria – Next Generation, New Ideas, vwh, 2014, pp. 183-198

2013 Märtin, C., Herdin, C., Engel, J.: Patterns and Models for Automated User Interface Construction – In Search of the Missing Links, in: M. Kurosu (Ed.), Human-Computer Interaction, Part I, HCII 2013, Las Vegas, U.S.A., LNCS 8004, pp. 401-410, Springer, Heidelberg (2013)

Engel, J., Herdin, C., Märtin, C., Forbrig, P.: Formal Pattern Specifications to Facilitate Semi-Automated User Interface Generation, in: M. Kurosu (Ed.), Human-Computer Interaction, Part I, HCII 2013, Las Vegas, U.S.A., LNCS 8004, pp. 300-309, Springer, Heidelberg (2013)

2012 Engel, J., Herdin, C., Märtin, C.: Pattern-oriented Modeling and Development of Interactive Information Systems, in: A. Frotschnig u. H. Raffaseder (Hrsg.), Proc. 5. Forum Medientechnik, St. Pölten, Austria – Next Generation, New Ideas, vwh, Hülsbusch, Glückstadt (2012), pp. 155-167

Engel, J., Herdin, C., Märtin, C.: Exploiting HCI Pattern Collections for User Interface Generation, Proc. Patterns 2012, Nice, France, IARIA 2012, pp. 36-44, available at http://www.thinkmind.org/index.php?view=article&articleid=patterns_2012_2_20_70024, last website call on September 23, 2012 (Best Paper Award at Patterns 2012)

2011	Engel, J., Herdin, C., Märtin, C.: A Task and Pattern-based Modeling Approach for Knowledge Sharing Systems. In: Forbrig, P., Dittmar, A. (Eds.) Designing Collaborative Activities, Proc. ECCE 2011, Rostock, Germany, August 24-26, 2011, pp. 275-276
	Engel, J., Märtin, C., Forbrig, P.: HCI Patterns as a Means to Transform Interactive User Interfaces to Diverse Contexts of Use, in: J.A. Jacko (Ed.): Human-Computer Interaction, Part I, HCII 2011, Orlando, U.S.A., LNCS 6761, Springer-Verlag Berlin Heidelberg 2011, pp. 204-213
	Engel, J., Märtin, C., Herdin, C.: Pattern-based User Interface Transformation for Knowledge Sharing Applications, in: Seissler, M. et al (Eds.) Proc. PEICS ´11, 2nd International Workshop on Pattern-Driven Engineering of Interactive Computing Systems, Pisa, Italy, June 13, 2011, pp. 5-8
2010	Märtin, C., Engel, J., Kaelber, C., Werner, I.: Using HCI-Patterns for Modeling and Design of Knowledge Sharing Systems, in: Forbrig, P., Günther, H. (Eds.): Proc. of BIR2010, Rostock, September, 29 - October, 01, LNBIP 64, Springer Verlag Berlin Heidelberg 2010, pp. 1-13
	Engel, J., Märtin, C.: A Pattern- and Model-Based Life-Cycle-Approach for Developing High-Quality Interactive Applications. Proc. of IHCI 2010, Freiburg, Germany, IADIS Press, 2010, pp. 59-67
	Engel, J., Märtin, C., Forbrig, P.: Tool-support for Pattern-based Generation of User Interfaces. Breiner, K. et al. (eds.): Proc. of the 1st Int. Workshop on Pattern-Driven Engineering of Interactive Computing Systems (PEICS ´10), Berlin, Germany, ACM International Conference Proceedings Series, 2010, pp. 24-27
2009	Engel, J., Märtin, C.: PaMGIS: A Framework for Pattern-Based Modeling and Generation of Interactive Systems, in J.A. Jacko (Ed.): Human-Computer Interaction, Part I, HCII 2009, LNCS 5610, Springer Verlag Berlin Heidelberg 2009, San Diego, CA, USA, July 19 – 24, 2009, pp. 826-835
2007	Buchholz, G., Engel, J., Märtin, C., Propp, S.: Model-based Usability Evaluation – Evaluation of Tool Support. Proceedings of HCI International, Beijing, China, 22-27 July, 2007, Springer LNCS 4450, pp. 1043-1052