Methodische Entwicklung von Mensch-Maschine-Schnittstellen unter Berücksichtigung nutzerzentrierter und modellbasierter Ansätze

Ron Becker, Christoph Ruckert, Enrico Tappert

Mensch-Maschine-Systemtechnik
Fraunhofer-Institut für Kommunikation, Informationsverarbeitung und Ergonomie
Neuenahrer Str. 20
53343 Wachtberg
ron.becker@fkie.fraunhofer.de
christoph.ruckert@fkie.fraunhofer.de
enrico.tappert@fkie.fraunhofer.de

Abstract: Der vorliegende Beitrag beschreibt eine Vorgehensweise zur modellbasierten und nutzerzentrierten Entwicklung von Mensch-Maschine-Schnittstellen. In aufeinanderfolgenden Phasen der Systementwicklung werden unterschiedliche Modelle angelegt bzw. weiterentwickelt, die jeweils als Ausgangsbasis für die nächste Phase benutzt werden. Die Vorgehensweise stützt sich auf die ISO 9241-210 (Human-centred design for interactive systems) und stellt den Nutzer in den Fokus aller Aktivitäten. Dabei wird vor allem Wert auf ein zentrales Projekttool gelegt, das die Ergebnisse zentral zusammenfasst, aufbereitet und so den Projektfortschritt überwachen lässt.

1 Einleitung

Die Mensch-Maschine-Schnittstelle (Human Machine Interface, HMI) ist die Schnittstelle eines zu entwickelnden Systems zum Benutzer. Diese Schnittstelle sollte intuitiv bedienbar sein und dem Benutzer alle Funktionen anbieten, die dieser zur Erledigung seiner Aufgabe benötigt. Ein gut gestaltetes HMI erhöht die Benutzbarkeit des Systems und führt dadurch zu einer größeren Akzeptanz durch den Benutzer.

Die hier vorgestellte Vorgehensweise hilft dabei, alle notwendigen Schritte bei der Erstellung eines ergonomisch gestalteten HMIs zu betrachten und die erarbeiteten Ergebnisse zu dokumentieren. Sie orientiert sich an der Vorgehensweise nach ISO 9241-210 [ISO10] (Nutzerzentrierte Gestaltung von interaktiven Systemen) ist jedoch um Verifikationsschritte in den einzelnen Phasen erweitert worden, damit die festgelegten (Teil-) Ergebnisse in Form von Modellen, Dokumenten und/oder Anforderungen direkt durch Experten verifiziert werden können (Abbildung 1). Dadurch können Fehlentwicklungen schnell erkannt und vermieden werden. Die erstellten Modelle werden mit Hilfe der UML 2.3 [OMG10] notiert.

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4

Durch die Festlegung auf eine bereits definierte formale Sprache wird der Interpretationsspielraum minimiert. Durch diese „gemeinsame Sprache“ werden auch die Arbeiten der am Projekt beteiligten Modellersteller (Anforderungsingenieure, Softwareingenieure, UI-Designer) verknüpft. Das Feedback der Benutzer (als ein wichtiges Instrument der ISO 9241-210) kann ebenfalls zu dieser Sprache erfolgen, da auf komplexe UML-Konstrukte verzichtet wird. Durch dieses gemeinsame Verständnis soll die Zusammenarbeit der einzelnen Projektbeteiligten gefördert und das Feedback der Benutzer qualitativ verbessert werden. Die erstellten Modelle, Dokumente und/oder Anforderungen sollten in einem zentralen Tool erstellt und verwaltet werden. Auf diese Art sind alle Ergebnisse zentral abgelegt und können bei Bedarf zielgruppengerecht exportiert werden, um die bereits erarbeiteten Ergebnisse zu präsentieren. Als Nebeneffekt führt diese zentrale Speicherung der Projektergebnisse zu einer besseren Übersicht über den bisher erreichten Projektfortschritt. Alle in diesem Artikel beschriebenen Modelle (mit Ausnahme der HMI-Gestaltungen aus Kapitel 4.2) können mit UML 2.3 konformen Tools erstellt werden. Als zentrales Projekttool für diesen Artikel (und andere Projekte) wurde der Enterprise Architect[1] eingesetzt (einschl. der HMI-Gestaltungen aus Kapitel 4.2).

Die in diesem Beitrag beschriebene Vorgehensweise beginnt mit der Analyse (Kapitel 2), die den Nutzungskontext (Kapitel 2.1) und die Aufgaben (Kapitel 2.2) identifiziert und daraus die funktionalen natürlich-sprachliche Anforderungen ableitet (Kapitel 2.3). Aus Abbildung 1 ist zu entnehmen, dass die Analyse die Aktivitäten 1 und 2 der ISO 9241-210 umfasst. Bereits nach der Aufgabenanalyse steht aber schon ein Modell zur Verfügung, dass sich zur Verifikation durch die Stakeholder[2] eignet. Gleiches gilt für das Ende der Analysephase.

In der Designphase (Kapitel 3) wird die Entwicklung eines Prototypen vorbereitet, indem die Interaktionen beschrieben und modelliert werden. Es wird ein Modell erzeugt, das schon vor Ablauf der Aktivität 3 aus der ISO 9241-210 zur Verifikation herangezogen werden kann.

Anschließend werden im Prototyping (Kapitel 4) Oberflächenlayouts erarbeitet und in einem iterativen Prozess interaktive Gestaltungslösungen entwickelt.

Der erstellte Prototyp wird anschließend von unterschiedlichen Benutzern evaluiert (Kapitel 5). Falls das Ergebnis der Evaluation positiv ausfällt, kann die Entwicklung des Systems angestoßen werden. Falls das Ergebnis der Evaluation jedoch negativ ist, werden je nach den gewonnenen Erkenntnissen einzelne Phasen der Vorgehensweise wiederholt.

Es sei außerdem darauf hingewiesen, dass die Vorgehensweise sowohl in verschiedenen Abstraktionsgraden (angefangen bei groben Aufgabenbeschreibung und Papierprototypen) iterativ durchlaufen werden kann, als auch separate Zyklen für einzelne Aufgaben durchführbar sind.

---

[1] http://www.sparxsystems.com/products/ea/index.html

[2] „individual or organization having a right, share, claim or interest in a system or in its possession of characteristics that meet their needs and expectations” [ISO10, 2.11]

ISBN 978-3-88579-286-4

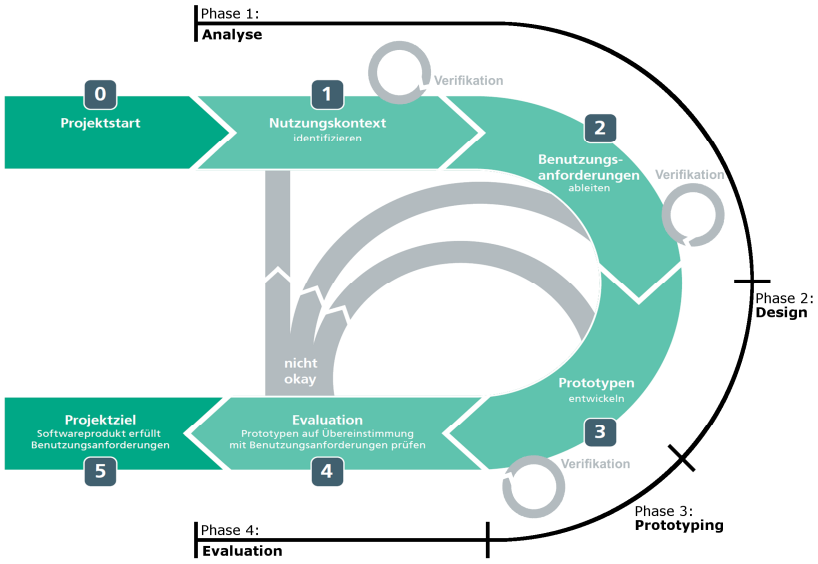

Abbildung 1: Vorgehensweise nach ISO 9241-210 mit Phasen und Verifikationsschritten erweitert

1.1 Beispiel

Für ein besseres Verständnis wird in diesem Beitrag ein durchgängiges Beispiel verwendet: „Es soll ein Online-Shop erstellt werden. Ein Kunde kann in dem Shop Produkte suchen und kaufen. Falls er ein Produkt kaufen möchte und bereits über ein Benutzerkonto verfügt, muss sich der Kunde in sein Benutzerkonto einloggen. Sollte der Kunde kein Benutzerkonto besitzen, kann er ein neues Benutzerkonto anlegen. Nach erfolgreichem Login kann er sich außerdem die letzten 50 Bestellungen anzeigen lassen.“

2 Analyse (Phase 1)

Eine strukturierte Analyse muss durchgeführt werden, damit der Benutzer steuern kann, was er steuern muss, bedienen kann, was er bedienen will, findet, was er benötigt, sieht, was er sehen muss, einstellen kann, was er einstellen will, usw.

Die Analyse ist eine zentrale Voraussetzung, um ein ergonomisch gestaltetes HMI zu entwickeln. Eine 1997 durchgeführte Studie ergab, dass etwa 60 Prozent aller Benutzbarkeitsprobleme darauf zurückgehen, dass der Dialogablauf nicht dem Arbeitsablauf entspricht [GH98]. Das heißt, dass das HMI nicht optimal an die zu lösende Aufgabe angepasst ist. Dabei sind Inkonsistenzen für 25 Prozent und eine generelle Unübersichtlichkeit der Dialogelemente für 15 Prozent der Probleme verantwortlich.

Die Ergebnisse, der vorliegenden Vorgehensweise werden in einer formal definierten Form festgehalten, damit alle Projektbeteiligten eine identische Sicht auf die Ergebnisse haben und es keinen Interpretationsspielraum gibt. Dazu werden Diagramme der UML oder selbst definierte Dokumente verwendet, die von Software- und Anforderungsingenieuren erstellt werden.

Im Laufe der Analyse wird eine gemeinsame Kommunikationsbasis erarbeitet und in einem Glossar verwaltet, so dass die verwendeten Begriffe eindeutig definiert sind.

2.1 Nutzungskontext analysieren

In einem ersten Schritt wird der Nutzungskontext analysiert. Dabei wird eine IST-Analyse durchgeführt, bei der der SOLL-Zustand der Schnittstelle berücksichtigt wird. Es wird eine Produktübersicht erstellt, die alle Leistungen, Aufgaben und Einschränkungen der Schnittstelle enthält. Außerdem werden die Akteure und externen Systeme identifiziert, die mit dem System interagieren. In dem konkreten Beispiel aus Kapitel 1.1 wären dies ein Kunde und eine Produktdatenbank. Die Ergebnisse werden in einer Prosabeschreibung des Nutzungskontextes zusammengefasst. Der Benutzer überprüft den Nutzungskontext, um zu gewährleisten, dass bei der Gestaltung der Schnittstelle alle notwendigen Funktionen und Leistungen berücksichtig werden.

2.2 Aufgabenanalyse durchführen

Das Ziel bei der Aufgabenanalyse besteht darin, zu erkennen, was der Benutzer mit dem Anwendungssystem erreichen möchte bzw. sollte, welche Strategien und Techniken er dabei benutzt und welche Informationen er für ihre Realisierung benötigt [Pr99]. Es werden die Aufgaben, die bei der Erstellung der Produktübersicht identifiziert wurden, in einem Anwendungsfalldiagramm modelliert. Dabei werden die einzelnen Anwendungsfälle mit den beteiligten Akteuren assoziiert. Außerdem werden Beziehungen zwischen einzelnen Anwendungsfällen modelliert, falls ein Anwendungsfall Teil eines weiteren Anwendungsfalls ist. Dabei ist darauf zu achten, ob diese Teilaufgabe in jedem Fall («include» Beziehung) oder nur unter bestimmten Voraussetzungen («extend» Beziehung) ausgeführt werden soll.

Zu jedem Anwendungsfall wird in der weiteren Bearbeitung zusätzlich ein Aktivitätsdiagramm erstellt, welches den genauen Ablauf und eventuelle Ausnahmen der Aufgabe visualisiert. Dabei kann durch Vergabe von Stereotypen modelliert werden, welche Akteure die einzelnen Aktionen innerhalb der Aufgabe durchführen.

Man beginnt mit einer zentralen Aufgabe, die als Anwendungsfall in ein Anwendungsfalldiagramm eingefügt und mit den beteiligten Akteuren verbunden wird. Für die Benennung des Anwendungsfalls sollte ein Nomen und ein Prozesswort (Verb) verwendet werden (z.B. „Passwort eingeben“), um später auf einfache Weise Anforderungen zu erzeugen. Anschließend wird eine textuelle Beschreibung hinzugefügt, die u.a. Akteure, Vorbedingungen, Ergebnisse, Ablaufbeschreibungen, etc. beinhaltet.

ISBN 978-3-88579-286-4

Die Ablaufbeschreibung besteht aus den essentiell benötigten Schritten (Essenzschritte) der Aufgabe. Setzt ein Essenzschritt eine Bedingung voraus, ist dieser eine Teilaufgabe des Anwendungsfalls, die in einem separaten Anwendungsfall modelliert wird und über eine «extend» Beziehung mit dem Anwendungsfall verbunden wird.

Sollten die Ablaufbeschreibung eines Anwendungsfall zu komplex werden (7 +/-2 Essenschritte [Mi56]), wird der Anwendungsfall in Teilaufgaben aufgeteilt und in separaten Anwendungsfällen modelliert. Diese Teilaufgaben, die von keiner Bedingung abhängen, werden auf jeden Fall ausgeführt. Um dies in dem Modell zu verdeutlichen, wird der Anwendungsfall der Teilaufgabe über ein «include» mit dem Ausgangsanwendungsfall in Beziehung gesetzt.

Sobald die Essenzschritte für einen Anwendungsfall aufgeschrieben worden sind, wird mit weiteren, noch nicht beschriebenen Anwendungsfällen analog verfahren. Falls ein Essenzschritt in mehreren Anwendungsfallbeschreibungen verwendet wird, wird dieser aus den Anwendungsfällen herausgezogen und mit diesen über «include»-Beziehungen verbunden. Sollte ein identifizierter Essenzschritt schon als separater Anwendungsfall modelliert worden sein, wird der zu beschreibende Anwendungsfall mit dem Anwendungsfall des Essenzschrittes in Beziehung («include») gesetzt.

Sind alle Anwendungsfälle für eine Aufgabe beschrieben, werden für weitere Aufgaben Anwendungsfälle erstellt und das Vorgehen in analoger Weise angewendet. Das Vorgehen endet, sobald alle Anwendungsfälle mit Anwendungsfallbeschreibungen versehen und die Aufgaben der Produktübersicht modelliert wurden. Das Ergebnis dieses Schritts ist ein vollständiges Anwendungsfalldiagramm. (Abbildung 2).

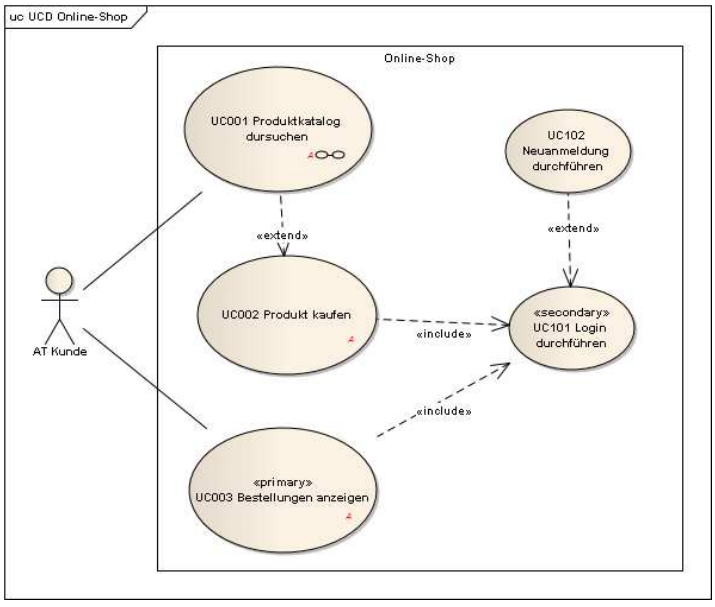

Abbildung 2: Anwendungsfallmodell des Online-Shop

ISBN 978-3-88579-286-4

Für eine bessere Übersicht werden die einzelnen Anwendungsfälle in Paketen zusammengefasst, die jeweils thematisch zusammenhängen. Anschließend wird zu jedem Anwendungsfall ein Aktivitätsdiagramm erstellt, das jeweils den Ablauf eines Anwendungsfalls mittels aufeinander folgender Aktionen weiter detailliert. Jeder Essenzschritt der Anwendungsfallbeschreibung ist durch min. eine Aktion modelliert.

2.3 Anforderungen ableiten

Sobald die Erstellung der Diagramme abgeschlossen ist, kann durch ein strukturiertes Vorgehen ein Großteil der funktionalen Anforderungen an das HMI erhoben werden. Durch bestimmte Schlüsselworte wie „muss“, „soll“ oder „kann“ wird eine rechtliche Verbindlichkeit für die natürlich-sprachlichen Anforderungen festgelegt, die von allen Lesern verstanden werden, unabhängig davon, ob diese UML-Kenntnisse besitzen oder nicht. Des Weiteren ist die UML „semi-formal“, sodass die Semantik nicht immer festgelegt ist. Für eine rechtliche Verbindlichkeit oder Qualitätssicherung ist eine eindeutige Notation zwingend erforderlich, sodass hier auf natürlich-sprachliche Anforderungen zurückgegriffen wird. Die hier erzeugten Anforderungen sind aus den o.g. Gründen nur als initialer Anforderungskatalog zu sehen.

1. Schritt: Aufbauen einer Kapitelstruktur

Für die Erstellung einer Kapitelstruktur wird die in der Aufgabenanalyse erstellte Paketeinteilung verwendet. Dabei liefern die Paketnamen die Kapitelüberschrift und Namen von Unterpaketen dementsprechend Titel von Unterkapiteln.

2. Schritt: Erstellen natürlich-sprachlicher Anforderungen für jeden Anwendungsfall

Zu jedem Anwendungsfall des Anwendungsfallmodells wird eine natürlich-sprachliche Anforderung erstellt, was sicherstellt, dass das System die Funktionalitäten für die Aufgaben und Teilaufgaben zur Verfügung stellt (rechtliche Verbindlichkeit durch das Schlüsselwort „muss“ [RS09]).

Das System <Name> muss <AWF> können.
↓
Das System Online-Shop muss den Produktkatalog durchsuchen können.

3. Schritt: Erstellen natürlich-sprachlicher Anforderungen für die Assoziation mit den Akteuren

In diesem Schritt werden Anforderungen erstellt, die festlegen, welche Akteure eine bestimmte (Teil-)Aufgabe durchführen können. Sie betreffen das HMI des Systems.

Das HMI zum <Name> muss <Akteur> die Möglichkeit bieten, <AWF>.
↓
Das HMI zum Online-Shop muss dem Benutzer die Möglichkeit
bieten, den Produktkatalog zu durchsuchen.

4. Schritt: Beschreiben / Erstellen von «include»-Beziehungen in natürlich-sprachlichen Anforderungen

Falls ein Anwendungsfall einen weiteren Anwendungsfall inkludiert, wird der zweite Anwendungsfall auf jeden Fall ausgeführt. In einem solchen Fall muss in den natürlich-sprachlichen Anforderungen festgehalten werden, zu welchem Zeitpunkt der inkludierte Anwendungsfall ausgeführt wird.

Sobald <Bedingung>, muss das HMI <includierter AWF>
↓
Sobald der Benutzer ein Produkt kaufen möchte, muss das HMI dem Benutzer die Möglichkeit bieten, einen Login durchzuführen.

5. Schritt: Beschreibung / Erstellen von «extend»-Beziehungen in natürlich-sprachlichen Anforderungen

Steht ein Anwendungsfall mit einem zweiten Anwendungsfall (Teilaufgabe) in einer «extend» Beziehung, wird der Anwendungsfall der Teilaufgabe nur unter bestimmten Bedingungen ausgeführt. Diese Bedingungen sind in den Essenzschritten festgehalten und müssen in die natürlich-sprachlichen Anforderungen übertragen werden.

Falls <Bedingung>, muss das HMI <Akteur> die Möglichkeit bieten,
<extendierter AWF>
↓
Falls der Benutzer einen Login durchführen möchte und über kein Benutzerkonto verfügt, muss das HMI dem Benutzer die Möglichkeit bieten, eine Neuanmeldung durchzuführen.

Für eine bessere Übersicht in den natürlich-sprachlichen Anforderungen sollten die Anforderungen aus Schritt 5 mit den in Schritt 3 erzeugten verlinkt werden.

6. Schritt: Erstellung der Anforderungen für die Aktionen der Aktivitätsdiagramme

In Schritt 6 erfolgt die Erstellung natürlich-sprachlicher Anforderungen für die Aktionen der Aktivitätsdiagramme, falls diese nicht schon in den Schritten 1 – 5 erstellt worden sind.

3 Design (Phase 2)

Die Designphase widmet sich der Erarbeitung eines Lösungsvorschlags unter bestimmten Rahmenbedingungen. Es wird in dieser Phase festgelegt, *wie* die in der Analyse herausgearbeiteten Anforderungen erfüllt werden können. Hauptsächlich werden hier Struktur- und Verhaltensmodelle aufgestellt. Strukturmodelle sind statische Modelle, die angeben, aus welchen Bestandteilen ein betrachteter Bereich besteht und in welcher Beziehung diese zu einander stehen. Verhaltensmodelle hingegen sind dynamische Modelle, die die Zusammenarbeit der einzelnen Bestandteile beschreiben.

ISBN 978-3-88579-286-4

Da die Entwicklung von Benutzungsschnittstellen als ingenieurswissenschaftlicher Prozess aufgefasst werden soll, ist der Inhalt der Designphase mit der Entwicklung anderer Softwareteile ähnlich. Das gilt auch für die bevorzugten UML-Diagramme (Klassen- und Sequenzdiagramme). Allerdings ist das Ziel statischer und dynamischer Modelle ein anderes. Mit einem statischen Modell werden das System und der Benutzer als Schnittstelle beschrieben. Mit einem dynamischen Modell werden die Interaktionen zwischen System und Benutzer spezifiziert.

Die Modelle in dieser Phase werden hauptsächlich von Softwareingenieuren erstellt. Lediglich am Ende der Phase kann ein UI-Designer bereits beratend tätig werden, um so auch schon Projektwissen aufzubauen. Der künstlerische Aspekt bei der Gestaltung eines HMIs kann und soll jedoch nicht wegrationalisiert werden. Er soll aber zu einem späteren Zeitpunkt (Phase 3) ausgeprägt werden, um auf einer stabilen (modellbasierten) Basis aufzubauen und um keine Realisierungsdetails vorwegzunehmen. Wie bei der Beschreibung von Phase 3 noch herausgearbeitet wird, kann die (künstlerische) Arbeit eines UI-Designers in die (ingenieurswissenschaftlichen) Modelle der Softwareingenieure eingebettet werden, sodass gemeinsame Dokumente einen projektweiten Überblick erlauben und eine direkte Ausimplementierung ermöglichen.

3.1 Interaktionsmöglichkeiten, -angebot und –umsetzung (statisches Modell)

Entsprechend dem Begriff „Benutzungsschnittstelle“ wird eine Schnittstelle geschaffen, über die der Benutzer das System handhabt. Dadurch entsteht eine Interaktion zwischen Benutzer und System. In Abhängigkeit von der Beschaffenheit des Systems und des Benutzers existiert eine Reihe von Interaktionsmöglichkeiten. Diese müssen jedoch nicht alle durch die Benutzungsschnittstelle unterstützt werden. So entsteht ein Interaktionsangebot, in dem nur die angebotenen / gewünschten Interaktionen aufgeführt sind. Letztendlich können die angebotenen Interaktionen auf verschiedene Weisen umgesetzt werden. Abbildung 3 demonstriert die Interaktionsebenen einer Interaktion.

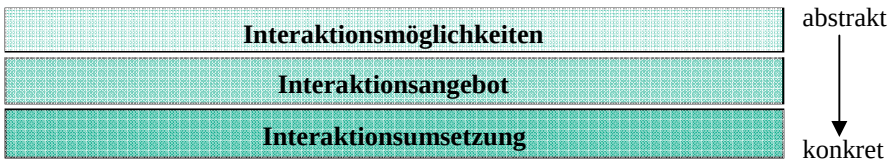

Abbildung 3: Abstraktionsebenen der Interaktion

Im statischen Modell werden diese Abstraktionsebenen modelliert, beginnend bei den Interaktionsmöglichkeiten (Beispiel siehe Abbildung 4). Dem damit verbundenen Aufwand stehen die Vorteile eines strukturierten Vorgehens gegenüber und schafft bei allen Stakeholdern ein Bewusstsein für die Komplexität des HMI-Designs, sowie das Bewusstsein, dass Änderungen bspw. an der Peripherie des Systems (im Nachhinein soll ein touchfähiger Bildschirm eingesetzt werden) eine Überarbeitung der Benutzungsschnittstelle erforderlich machen, um eine ergonomische Bedienung zu gewährleisten.

ISBN 978-3-88579-286-4

Um die Interaktionsmöglichkeiten zu erhalten, werden zuerst die Schnittstellen vom System zum Benutzer und vom Benutzer zum System definiert. Die einzelnen Bestandteile der Schnittstelle werden Schnittstellen-Knoten genannt. Aus Sicht des Benutzers bilden seine Schnittstellen die Sinnesorgane (Auge, etc.) und damit zusammenhängende Extremitäten (Hand, etc.). Die Schnittstelle im Hinblick auf das System ist hingegen durch Ein- / Ausgabegeräte (Maus, Bildschirm etc.) definiert.

Im nächsten Schritt werden die Schnittstellen-Knoten durch gerichtete Kanten verbunden, deren Richtung sich auf den Informationsfluss bezieht. Durch die Kanten wird definiert, welcher Schnittstellen-Knoten des Benutzers mit einem des Systems interagieren kann und vice versa. So entsteht ein Graph mit Paaren von Schnittstellen-Knoten, deren Vereinigung die Interaktionsmöglichkeiten darstellen.

Bei der Aufstellung der Interaktionsmöglichkeiten können auch Besonderheiten wie bspw. das Tragen von Handschuhen berücksichtigt werden, sodass in solchen Fällen bspw. keine Kanten zu bestimmten Touchbildschirmen gezogen werden dürfen. Ebenso können Besonderheiten der Interaktionsmöglichkeiten körperlich eingeschränkter Benutzer berücksichtigt werden.

Für die Notation der Interaktionsmöglichkeiten bietet sich eines der Strukturdiagramme der UML an, da diese eine statische Struktur von Objekten in einem System aufzeigen (vgl. [OMG10, S. 704]). Wir haben uns für das Klassendiagramm entschieden, weil es zum einen das bekannteste Strukturdiagramm sein dürfte und zum anderen als „Kern der gesamten Modellierungssprache“ [RQZ07, S. 101] angesehen werden kann. Da Klassen eine „Definition der Attribute, Operationen und der Semantik für eine Menge von Objekten“ [Oe09, S. 274] festlegen, eignen sich diese als Repräsentation der Schnittstellen-Knoten, für die es in der „realen Welt“ entsprechende Instanzen gibt. Zur besseren Lesbarkeit und um die Schnittstellen-Knoten von anderen Klassen abzuheben, werden diese mit einem passenden Stereotyp versehen. Dieser erlaubt die Erweiterung einer existierenden Metaklasse, um domainspezifische Terminologie oder eigene Notationen (neben der ursprünglichen) zu verwenden (vgl. [RQZ07, S. 514]). Die einzelnen Schnittstellen-Knoten werden über gerichtete Assoziationen miteinander verbunden, wodurch entsprechende Tupel gebildet werden (vgl. [OMG10, S. 38]). Um deutlich zu machen, dass die Assoziationen den Informationsfluss darstellen, werden diese mit dem Schlüsselwort «flow» gekennzeichnet (vgl. [RQZ07, S. 159f]). Das obere Drittel in Abbildung 4 zeigt die Interaktionsmöglichkeiten am Beispiel des Online-Shops.

Aufbauend auf den Interaktionsmöglichkeiten wird ein konkreteres Interaktionsangebot erstellt, das aus Interaktionsklassen besteht, die aus den Paaren der Schnittstellen-Knoten abgeleitet werden können. Aus dem Schnittstellen-Knoten-Paar „Hand → QWERTZ-Tastatur“ kann bspw. die Interaktionsklasse „Texteingaben“ abgeleitet werden. Es können auch durchaus verschiedene Schnittstellen-Knoten-Paare die gleiche Interaktionsklasse bedingen. Aus welchen Paaren sich welche Interaktionsklassen ergeben, auch im Hinblick auf nicht-grafische Benutzungsschnittstellen, ist noch nicht abschließend untersucht.

ISBN 978-3-88579-286-4

Da eine Interaktionsklasse eine Menge von Eigenschaften definiert, die nicht zu einer der Schnittstellen-Knoten, sondern zu der dazwischen liegenden Assoziation gehört, werden diese als Assoziationsklassen im Klassendiagramm der Interaktionsmöglichkeiten notieren (vgl. [RQZ07, S. 151]). Außerdem wird jede Assoziationsklasse, dessen zugehörige Assoziation einen Informationsfluss vom System zum Benutzer repräsentiert, mit dem Stereotyp «out» versehen. Für Assoziationen mit entgegengesetztem Informationsfluss wird die Assoziationsklasse mit dem Stereotyp «in» markiert. Die Angabe dieser Stereotypen ist zwar redundant zu der Richtung der Assoziationen, erhöht allerdings die Übersichtlichkeit (siehe Abbildung 4).

Auf Basis des Interaktionsangebots kann eine Interaktionsumsetzung erarbeitet werden, die aus konkreten HMI-Elementen besteht. Manche HMI-Elemente sind nur durch die Kombination verschiedener Interaktionsklassen möglich (z.B. ergeben sich „Schaltflächen“ aus den Interaktionsklassen „Cursorsteuerung“ und „Grafische Ausgabe“). Es kann nun wiederum je nach Schnittstellen-Knoten-Paar, aus dem eine Interaktionsklasse hervorging, eine Bewertung des abgeleiteten HMI-Elements vorgenommen werden.

Es ist auch vorstellbar, dass die Ableitungen der Interaktionsklassen und später der HMI-Elemente wiederum in einem Graph repräsentiert werden, dessen Kanten mit der oben angesprochenen Bewertung gewichtet werden. So entsteht eine Art Landkarte, die angibt, welche Interaktionsklasse von welchen Interaktionsmöglichkeiten am besten und welches HMI-Element am besten von welcher Interaktionsklasse unterstützt wird. Ebenso können Redundanzen erkannt bzw. falls notwendig ergänzt werden.

Das Klassendiagramm des Interaktionsangebots wird um die HMI-Elemente erweitert, die als Klassen notiert werden, da diese zu einem späteren Zeitpunkt durch konkrete Klassen einer Grafikbibliothek ersetzt werden können. Ein HMI-Element wird mit einer Assoziation zu jeder Interaktionsklasse verbunden, aus der sie hervorgeht. Für die spätere Implementierung können die Assoziationen durch Stereotype genauer spezifiziert werden. Die Abhängigkeitsbeziehung drückt bspw. aus, dass das HMI-Element die entsprechende Interaktionsklasse benötigt (vgl. [OMG10, S. 140]), jedoch keine Teile davon implementiert. Die Realisierungsbeziehung hingegen gibt an, dass Teile einer Interaktionsklasse durch ein konkretes HMI-Element implementiert wird (vgl. [OMG10, S. 132]). Abbildung 4 zeigt das Klassendiagramm für die Interaktionsumsetzung am Beispiel des Online-Shops.

Das statische Modell dient als Komponentenpool, aus dem in der folgenden Interaktionsbeschreibung (Kapitel 3.2) bzw. HMI-Gestaltung (Kapitel 3.3) geschöpft werden kann. Es ist in dieser Phase noch abstrakt gehalten, um programmiersprachenunabhängig zu bleiben. Die konkrete Programmiersprache wird erst in Phase 3 festgelegt.

ISBN 978-3-88579-286-4

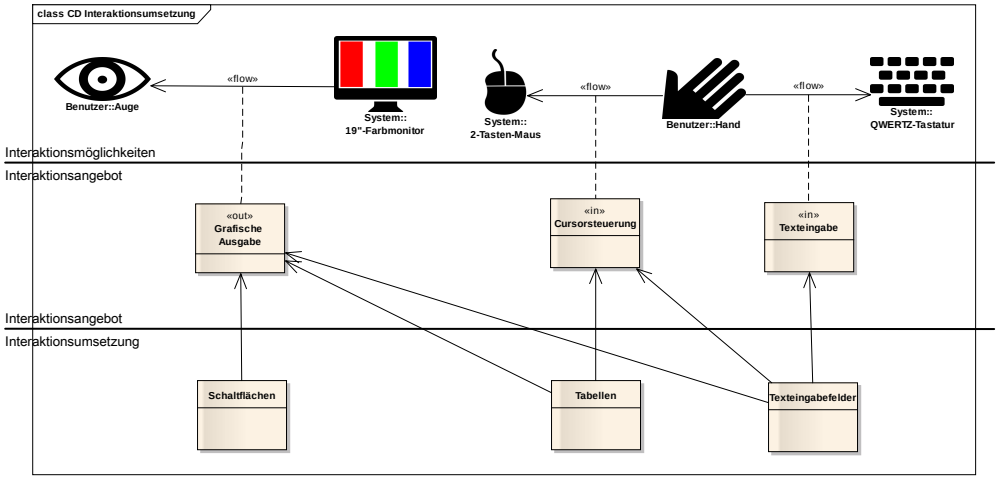

Abbildung 4: Interaktionsmöglichkeiten, -angebot und –umsetzung für den Online-Shop

3.2 Interaktionsbeschreibung (dynamisches Modell)

Die Interaktionsbeschreibung veranschaulicht die Interaktion zwischen Benutzer und System frühzeitig, aber noch auf einer abstrakten Ebene. Ziel dieses Schritts ist die Erstellung eines Interaktionsmodells, das zum einen beschreibt was das HMI dem Benutzer und zu anderen was das System dem HMI anbieten muss. Das Ergebnis sind zwei Schnittstellenbeschreibungen.

Für die Notation bietet sich eines der Verhaltensdiagramme der UML an, da diese das Verhalten der Objekte in einem System aufzeigen (vgl. [OMG10, S. 704]). In [HSW02] werden hierfür Aktivitätsdiagramme vorgeschlagen, da dort jedoch die Abfolge von Nachrichten, als die Nachrichten selbst oder deren Kommunikationspartner von Interesse sind (vgl. [OMG10, S. 303]), eignet sie sich für die Interaktionsbeschreibung nur bedingt. Das Sequenzdiagramm fokussiert hingegen auf die beiden zuletzt genannten Punkte und verspricht daher eine bessere Visualisierung der Schnittstellen (vgl. [RQZ07, S. 397]). Für jeden Anwendungsfall wird ein Sequenzdiagramm erstellt, das in der Regel aus 3 Lebenslinien besteht. Die erste Lebenslinie stellt den involvierten Benutzer dar, die zweite das HMI und die dritte das System. Sind mehrere Benutzer beteiligt, so kommt für jeden weiteren Benutzer eine weitere Lebenslinie hinzu. Analog wird verfahren, wenn das HMI verschiedene Systeme ansprechen muss. Beginnend mit einer Benutzeraktion wird der Ablauf des Standardfalls notiert. Die Nachrichten zwischen den einzelnen Lebenslinien, sind so weit wie möglich spezifiziert. Sollten Parametertypen und / oder Rückgabetypen noch nicht bekannt sein, weil diese von der Interaktionsumsetzung abhängen, so werden diese leer gelassen oder mit „...“ oder „?“ besetzt. So ergibt sich ein Überblick, welche Teile der Schnittstelle von der Interaktionsumsetzung abhängig (und somit auch bei Änderungen dergleichen betroffen) sind.

ISBN 978-3-88579-286-4

Ausnahme- oder Alternativabläufe sollten ebenfalls modelliert werden. Hierbei kann man jedoch unterscheiden, ob auf eine allgemeine Ausnahmebehandlung (Fehlermeldung an einer zentralen Stelle des HMI-(Fragments)) zurückgegriffen werden soll (die zur Reduzierung der Dokumente nur einmal zentral modelliert werden sollte) oder ob die Ausnahmebehandlungen (was das HMI betrifft) speziell an den Anwendungsfalls angepasst werden muss. In beiden Fällen jedoch sollte die Verknüpfung zum Alternativ-/Ausnahmeablauf aus dem Sequenzdiagramm des Standardfalls heraus erfolgen und ggf. nach Bearbeitung der Alternative/Ausnahme dorthin zurückkehren. Für die Alternativ-/Ausnahmeabläufe sollten dann separate Sequenzdiagramme nach dem oben beschriebenen Verfahren vom Beginn bis zum Ende der Alternative/Ausnahme erstellt werden. Die Verknüpfung aus dem Standardfall kann mit Hilfe einer Interaktionsreferenz (vgl. [RQZ07, S. 460]) erfolgen.

3.3 HMI-Gestaltung ableiten

Das dynamische Modell lässt sich nun mit Elementen des statischen Modells anreichern, um so eine (abstrakte) HMI-Gestaltungen abzuleiten.

Dabei werden für die Nachrichten vom Benutzer zum HMI und umgekehrt Elemente aus der Interaktionsumsetzung ausgewählt, die zur Anzeige oder Bedienung verwendet werden sollen und mit ihrer Anzahl notiert. Da die UML kein Element kennt, das an Nachrichten geheftet werden kann, um die grafische Aufbereitung der Nachricht zu visualisieren, wird hier das (mit beliebigen UML-Modellelemente kombinierbare [Oe09, S. 297]) Notiz-Element verwendet (vgl. Abbildung 5).

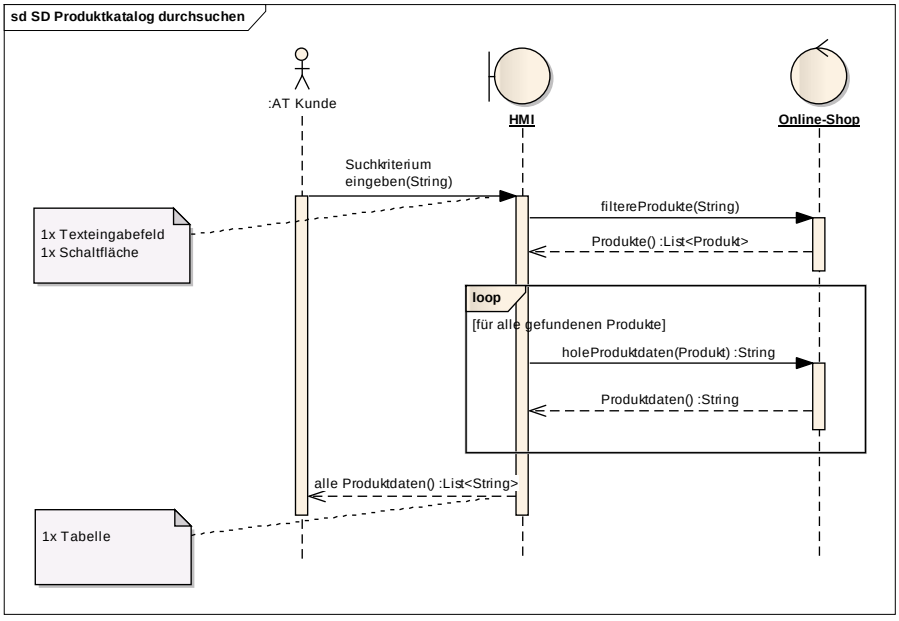

Abbildung 5: Abstrakte HMI-Gestaltung für den Anwendungsfall „Produktkatalog durchsuchen“

ISBN 978-3-88579-286-4

Die noch fehlenden Parameter bzw. Rückgabetypen in den Sequenzdiagrammen können danach entsprechend belegt werden. Die Position und Größe bzw. Anordnung der HMI-Elemente wird in diesem Schritt noch nicht modelliert. Somit ist das Ergebnis der Design-Phase ein generisches Modell, das erst in den folgenden Schritten zu einem konkreten HMI ausgearbeitet wird. Damit soll die Stabilität der Designdokumente erhöht und der Änderungsaufwand minimiert werden. Trotzdem ergibt sich ein erster (abstrakter) Überblick über die Interkation, sowie eine Aufteilung des Gesamt-HMIs in kleinere Komponenten (im folgenden HMI-Fragmente genannt). In den folgenden Schritten werden diese Komponenten nun sukzessive konkretisiert. Die komponentenweise Realisierung ermöglicht es die Komplexität des Gesamt-HMIs aufzuteilen und so besser beherrschbar zu machen. Es wird außerdem berücksichtigt, dass die Bewertung des HMIs erst in der Phase 4 (Evaluation) erfolgt und Änderungen daher noch sehr wahrscheinlich sind.

4 Prototyping (Phase 3)

4.1 Grobentwurf

Für ein komplettes HMI fehlen noch einige globale (also von einem bestimmten Anwendungsfall unabhängige) Entscheidungen. So muss z.B. festgelegt werden, ob Fenster- und / oder Popupmenüs erlaubt sind, ob es sich um eine Web- oder Desktop-Applikation handelt, ob ein Styleguide verwendet werden soll, wie viel Platz für das HMI zur Verfügung steht, wie die einzelnen Anwendungsfälle gestartet werden sollen (Navigation) oder ob bestimmte Frameworks (z.B. QT[3] oder GTK[4]) verwendet werden sollen, die Einschränkungen mit sich bringen. Diese Entscheidungen müssen vor der Erstellung des Prototypen geklärt werden.

4.2 Feinentwurf

Nachdem die Aufgaben definiert wurde (Analyse) und die HMI-Elemente, die zur Lösung eingesetzt werden sollen (Design), sowie die Rahmenbedingungen (Grobentwurf) feststehen, können konkrete HMI-Fragmente erstellt werden.

Ein HMI-Fragment bündelt die HMI-Elemente, die während der Designphase in den HMI-Gestaltungen festgelegt wurden. Als Notation bietet sich eine an die Sequenzdiagramme der UML angelehnte Form an. Als Vorlage werden die Sequenzdiagramme aus der Designphase herangezogen, wobei Nachrichten an / vom HMI mit konkreten Abbildungen ergänzt werden (vgl. Abbildung 6). Die Sequenzdiagramme der UML eignen sich für diesen Schritt besonders, weil sie nicht auf den Ablauf fokussieren (so wie Aktivitätsdiagramme), sondern die Kommunikation samt der Kommunikationspartner in den Mittelpunkt stellen.

[3] http://qt.nokia.com/
[4] http://www.gtk.org/

ISBN 978-3-88579-286-4

Sollte das zu entwickelnde HMI aus mehreren Teilen bestehen (bspw. einer Oberfläche für einen Bildschirm und einer für einen Touchbildschirm), so können diese in den Sequenzdiagrammen als zwei Instanzen mit jeweils eigener Lebenslinie modelliert werden. Wichtig ist hier jedoch nicht zu detailliert zu werden. Ob die Modellierung aller Peripheriegeräte als eigene Instanzen zielführend ist, kann mangels ausreichend praktischer Erfahrungen noch nicht beurteilt werden. Bilden sie hingegen ebenfalls einen zu entwickelnden Teil des HMIs, so sollten sie auf jeden Fall als eigene Instanzen modelliert werden. Auch unter dem Aspekt der Teamarbeit bietet sich die Darstellungsform in Sequenzdiagrammen an, weil in diesem Fall mehrere Akteurinstanzen modelliert werden können.

Zu jeder HMI-Instanz wird dessen Zustand grafisch dargestellt und in die Sequenzdiagramme eingearbeitet. Dafür werden die Notiz-Elemente durch konkrete Bilder des HMIs ersetzt. Da, wie bereits oben erwähnt, ein UML-Element für die grafische Aufbereitung von Nachrichten fehlt, müssen die Bilder als von der UML unabhängige (jedoch von vielen Tools unterstütze) Bild-Elemente eingefügt werden. Spätestens an diesem Punkt erfolgt nun die Einbindung der UI-Designer. Das konkrete HMI wird dazu in einem Grafikeditor (oder sonstigem UI-Design-Tool) erarbeitet, dem im Idealfall die Interaktionsbeschreibungen vorliegen, sodass es nur die zur Verfügung stehenden HMI-Elemente anbietet. Der UI-Designer legt seinen Entwurf ab, der daraufhin in das Sequenzdiagramm übernommen wird. So kann man auf elegante Weise die Modellierungsprozesse der Softwareingenieure und UI-Designer verknüpfen.

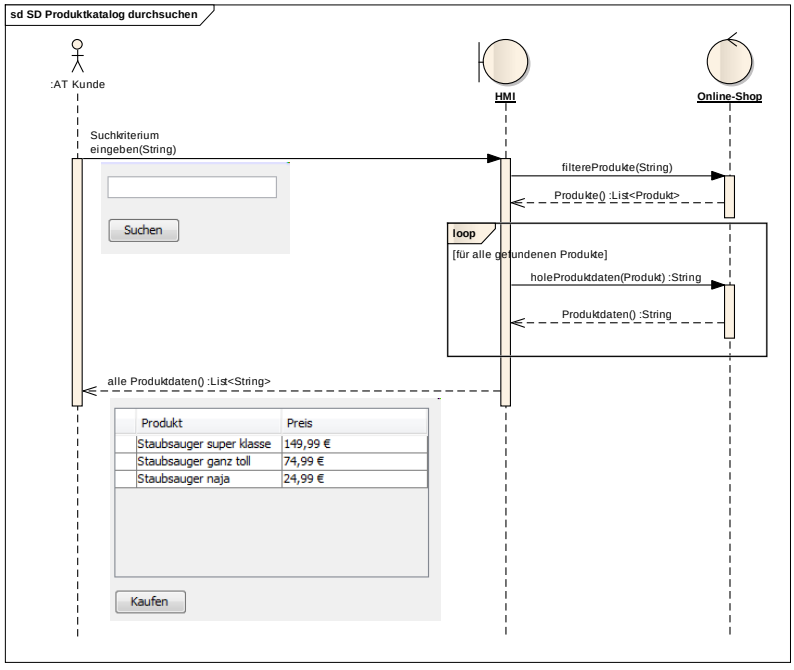

Abbildung 6: Konkrete HMI-Gestaltung für den Anwendungsfall „Produktkatalog durchsuchen“

ISBN 978-3-88579-286-4

Ein Tool, das dieses Vorgehen unterstützt müsste also die Palette für den UI-Designer in Abhängigkeit der zur Verfügung stehenden HMI-Elemente reduzieren und das erstellte HMI in die Sequenzdiagramme einpflegen. Die bisher verfügbaren Tools bieten diese Unterstützung jedoch nicht an, sodass die beschriebenen Vorgänge manuell durchgeführt werden müssen.

Ein weiterer Vorteil dieser Herangehensweise (der allerdings noch weiter untersucht werden muss) liegt in der automatisierten Erstellung von HMI-Prototypen, die mit Hilfe der um HMIs angereicherten Sequenzdiagramme abgeleitet werden können (siehe Kapitel 4.3). Diese automatisch generierten Prototypen könnten sowohl Anzeige, als auch Bedienung verdeutlichen, um wiederum frühzeitig Feedback durch die (zukünftigen) Benutzer zu erhalten. Am Ende des Feinentwurfs liegt zudem eine Beschreibung des HMIs vor, die eine direkte Ausimplementierung ermöglicht.

4.3 Implementierung

Die Implementierung im Rahmen des Prototypings beschränkt sich auf die Erstellung eines interaktiven Oberflächenvorschlags. Dazu können bereits eine konkrete Programmiersprache oder entsprechende Prototyping-Tools verwendet werden. Neben der reinen grafischen Darstellung sollte der Prototyp aber auch die Interaktionen zwischen Benutzer und System verdeutlichen. Er wird für die Evaluation in der folgende Phase verwendet.

5 Evaluation (Phase 4)

In einer Evaluation wird der erzeugte Prototyp bspw. nach den Grundsätzen der Dialoggestaltung [ISO06] durch mehrere Probanden getestet. Die Erfahrungen die durch diese Evaluation gewonnen werden bilden die Grundlage für eine Verbesserung des Systems.

6 Ausblick

Bei der Erarbeitung der Vorgehensweise haben sich weitere Themengebiete eröffnet. Die Definition der Interaktionsklassen ist ein weiteres Feld, in dem Forschungsarbeit zu leisten ist. Die Ableitung der Interaktionsklassen aus den Interaktionsmöglichkeiten, sowie die logische Strukturierung dieser, wird die modellbasierte Entwicklung von Benutzungsschnittstellen unterstützen. Ebenso müssten die HMI-Elemente gesammelt und entsprechend den Interaktionsklassen zugeordnet werden.

Aufgrund der oben beschriebenen Einteilungen könnten Studien zu der Bewertung einzelner HMI-Elemente in Abhängigkeit von den Schnittstellen-Knoten und den zu bewältigenden Aufgaben durchgeführt werden.

ISBN 978-3-88579-286-4

Weiterhin kann diese Vorgehensweise bei der Modellierung benutzt werden, um in Zukunft die Bewertung verschiedener HMI-Elemente auch in Abhängigkeit von externen Einflüssen (z.B. Bewegungseinflüsse bei Seegang) festzuhalten.

Es wird außerdem empfohlen ein neues UML-Element einzuführen, das mit dem Metaelement *Message* assoziiert werden kann (ähnlich, wie Assoziationsklassen). Dieses Element sollte einen Namen tragen können (also von *NamedElement* abgeleitet werden) und dazu Attribute aufnehmen können, zu denen eine Anzahl vermerkt werden kann. In einem dritten Bereich müsste außerdem eine grafische Repräsentation abgelegt werden können. Dieses Element kann dann das (nicht UML 2.3 konforme) Bild-Element aus Kapitel 4.2 ablösen und auch bereits für die Modellierungsaktivitäten in Kapitel 3.3 eingesetzt werden, um das semantisch schwache Notizelement abzulösen.

Ein die Vorgehensweise unterstützendes, den Benutzer leitendes UML Werkzeug, das ggf. auch eine automatisierte Prototypengenerierung beinhaltet ist außerdem zukünftige Entwicklungsarbeit.

Literaturverzeichnis

[GH98] Geis, T., Hartwig, R.: Auf die Finger geschaut – Neue ISO-Norm für benutzergerechte interaktive Systeme, c´t magazin, 14/98, S. 168-171, 1998.

[HSW02] Homrighausen, A., Six, H.-W., Winter, M.: Round-Trip Prototyping Based on Integrated Functional and User Interface Requirements Specifications, Requirements Engineering 6(1), Springer, 2002

[ISO06] ISO: DIN EN ISO 9241-110:2006 – Ergonomie der Mensch-System-Interaktion – Teil 110: Grundsätze der Dialoggestaltung. ISO (International Organization for Standardization), 2006. – (ISO 9241-110:2006) Deutsche Fassung EN ISO 9241-110:2006

[ISO10] ISO: ISO 9241-210:2010(E) – Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems. ISO (International Organization for Standardization), 2010

[Mi56] Miller, G. A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information, Psychological Review 63 (2): 343–355, 1956.

[Oe09] Oestereich, B.: Analyse und Design mit UML 2.3. Oldenbourg Verlag, München, 2009

[OMG10] OMG (Hrsg.): OMG Unified Modeling Language (OMG UML) Superstructure Version 2.3. 2010, http://www.omg.org/spec/UML/2.3/Superstructure/PDF/ (26.04.11)

[Pr99] Preim, B.: Entwicklung interaktiver Systeme, Grundlagen, Fallbeispiele und innovative Anwendungsfelder, Springer-Verlag Berlin Heidelberg New York, 1999.

[RQZ07] Rupp, C., Queins, S., Zengler. B.: UML 2 Glasklar. 3. Auflage, Carl Hanser Verlag, München, 2007

[RS09] Rupp, C., Sophist Group: Requirements-Engineering und –Management, Carl Hanser Verlag, München Wien, 2009

ISBN 978-3-88579-286-4