UNIVERSITÄT KONSTANZ

Fakultät Informatik und Informationswissenschaften
Master-Studiengang Information Engineering

MASTERARBEIT

Konzeption eines Unterstützungswerkzeugs für die Gestaltung ergonomischer Benutzerschnittstellen am Beispiel der Dr. Ing. h. c. F. Porsche AG

zur Erlangung des akademischen Grades
Master of Science (M. Sc.)

Bearbeiter:	Tobias Schwarz
Matr.-Nr.	01/604203
Gutachter:	Prof. Dr. Harald Reiterer
	Prof. Dr. Rainer Kuhlen
Betreuer:	M. Sc. Thomas Memmel
Einreichung:	03. September 2007

Lehrstuhl für Mensch-Computer-Interaktion

Sperrklausel

Die vorliegende Masterarbeit enthält zum Teil Informationen, die nicht für die Öffentlichkeit bestimmt sind. Der Inhalt darf daher nur mit der ausdrücklichen schriftlichen Genehmigung des Verfassers und der Dr. Ing. h. c. F. Porsche AG an Dritte weitergegeben werden.

PORSCHE

Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen als solche gekennzeichnet habe.

Konstanz, den 02.09.2007

Tobias Schwarz

Abstract

As the development periods of service panels or service systems are getting shorter and as the stress of competition increases, the need of supporting tools for ergonomic Human-Interface-Devices becomes much more important. Pressure is also created by raised norms and standards for automobiles, which have to be considered by today's developers. The aim of this thesis is to provide a module that supports the entire development process for service panels within the Porsche corporation. At present, existing tools cover the configuration of service panels only partly and not through the entire process of development, as a detailed analysis has shown.

Hence a concept for the supportive module *PoCKET* (**Po**rsche **C**onsulting and **K**nowledge **E**ngineering **T**ool) was developed within this thesis. It has been integrated in a tool-support chain, which is elaborately described in Bock (2007) and Heilig (2007). On the one hand PoCKET is a system that supports developers during the configuration of service panels in the dimensions interface, information and interaction design. On the other hand it provides a knowledge management system in which collaborative evolved information can be stored and accessed.

In order to develop a system that perfectly fits the HMI development process of Porsche, a in depth analysis of the development process has been accomplished. An interview in type of a questionnaire has been conducted quantitative as well as qualitative to determine the developer's requirements as detailed as possible. Therefore these results have been the initial point for an adequate configuration of the new supportive module. The consideration of existing tools for the configuration of ergonomic user interfaces has shown that these do not fit the particular industrial requirements of Porsche. For that reason they can only contribute as a connecting factor to this thesis. The aim of this thesis is to provide a component, which supports the HMI configuration process of the Porsche corporation entirely.Hereby the focus of this work lies on the support of specific process periods as well as the employment incross-project contexts. The surplus of the supportive module *PoCKET* is shown by considering the DATech testing manual for usability development processes. Thus an increasing quality of the HMI development process can be justified by the application of this module.

Kurzfassung

Die immer kürzeren Entwicklungszeiten von Bediensystemen und der steigende Wettbewerbsdruck führen dazu, dass Werkzeuge zur Gestaltung von ergonomischen Benutzerschnittstellen eine immer größere Bedeutung erlangen. Aufgrund der stetig zunehmenden Normen und Standards, die im Fahrzeug beachtet werden müssen, wächst der Druck auf die Entwickler bei der Berücksichtigung dieser Vorgaben.
Ziel dieser Arbeit ist es, den Entwicklern der Porsche AG eine Komponente bereitzustellen, die sie während des gesamten Entwicklungsprozesses unterstützt. Derzeit existieren zwar einige Werkzeuge, die Entwickler bei der Bediensystemgestaltung unterstützen, aber wie sich nach eingehender Analyse zeigte, nicht während des ganzen Entwicklungsprozesses bei der Porsche AG. Um diesem Aspekt entgegen zu wirken, wurde im Rahmen dieser Arbeit die Unterstützungskomponente PoCKET (**Po**rsche **C**onsulting and **K**nowledge Engineering **T**ool) konzipiert. Diese Komponente wurde in eine Werkzeugkette integriert, die in den Arbeiten von Bock (2007) und Heilig (2007) ausführlich beschrieben wird. *PoCKET* ist ein System, das zum einen die Entwickler bei der Bediensystemgestaltung in den Dimensionen *Oberflächen*, *Informations-* und *Interaktionsdesign* unterstützt und zum anderen ein Wissensmanagementsystem bereitstellt, in dem kollektive Informationen abgelegt werden können. Um das System optimal an den HMI-Entwicklungsprozess der Porsche AG anzupassen, wurde eine eingehende Analyse des Entwicklungsprozesses durchgeführt. Die Befragungsmethode in Form eines Fragebogens mit Interview war sowohl quantitativ als auch qualitativ, um die Anforderungen der Entwickler möglichst genau in Erfahrung zu bringen. Dadurch konnten die Ergebnisse der Befragung als Ausgangspunkt für eine den Anforderungen der Entwickler entsprechende Gestaltung der Unterstützungskomponente dienen. Bei der Betrachtung bestehender Werkzeuge zur Gestaltung ergonomischer Benutzerschnittstellen wurde festgestellt, dass diese den speziellen industriellen Anforderungen der Porsche AG nicht gerecht werden. Sie konnten daher lediglich als Anknüpfungspunkte für das Konzept dieser Arbeit dienen.
Im Rahmen der vorliegenden Arbeit soll nun ein Ansatz vorgestellt werden, der speziell den Anforderungen des HMI-Entwicklungsprozesses der Porsche AG gerecht wird. Dabei steht die Unterstützung in den Prozessphasen sowie eine projektübergreifende Unterstützung im Mittelpunkt. Der Mehrwert der Unterstützungskomponente *PoCKET* wird anhand des DATech-Prüfhandbuches für Usability-Entwicklungsprozesse beleuchtet. Somit kann durch den Einsatz dieser Komponente die steigende Qualität des HMI-Entwicklungsprozesses begründet werden.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Ausgangssituation 1
1.2 Motivation 2
1.3 Einordnung und Abgrenzung der Arbeit 4
1.4 Zielsetzung der Arbeit 5
1.5 Aufbau der Arbeit 6

2 Analyse des aktuellen Porsche AG HMI-Entwicklungsprozesses **7**

2.1 Zielsetzung der Analyse 7
2.2 Gang der Untersuchung 8
2.3 Ergebnisse der Untersuchung 10
2.4 Anforderungen an die Werkzeugunterstützung 16
2.4.1 Anforderungsprofil der Beratungskomponente 19
2.4.2 Anforderungsprofil der Wissenskomponente 19

3 Grundlagen **21**

3.1 Usability-Engineering-Prozesse 21
3.1.1 Der Usability-Engineering-Lifecycle 22
3.1.2 Contextual-Design 22
3.1.3 Usage-Centred-Design 23
3.1.4 Useware-Entwicklungsprozess 23
3.1.5 Erkenntnisse aus den Entwicklungsprozessen 24
3.2 Qualitätssicherung für Prozesse und Produkte 25
3.2.1 Prozessqualität 26
3.2.2 Produktqualität 27
3.3 Benutzerunterstützung in Form von Hilfesystemen 30
3.3.1 Was sind Hilfesysteme? 30
3.3.2 Klassifikation von Hilfesystemen 30
3.3.3 Formen von Hilfesystemen 34
3.4 Wissensmanagement 35
3.4.1 Was ist Wissen? 35
3.4.2 Wissen als entscheidender Wettbewerbsfaktor 40
3.4.3 Barrieren beim Wissensmanagement im Entwicklungsprozess 40

4 Werkzeuge zur Unterstützung von Usability-Entwicklungsprozessen 43
4.1 Werkzeuge zur Gestaltung von ergonomischen Benutzeroberflächen 43
4.1.1 GENIUS 44
4.1.2 JANUS 45
4.1.3 MUSE II 45
4.1.4 EXPOSE 46
4.1.5 ProUSE 47
4.1.6 ERGO-CONCEPTOR 48
4.1.7 Gegenüberstellung der Werkzeuge 48
4.2 Werkzeuge zur automatischen Bewertung von Benutzeroberflächen 50
4.2.1 ErgoVal 50
4.2.2 KRI/AG 50
4.2.3 Sherlock 51
4.2.4 KWARESMI 51
4.2.5 ISO 9241-Evaluator 52
4.2.6 AIDE 53
4.2.7 Gegenüberstellung der Werkzeuge 53
4.3 Beratungswerkzeuge zur ergonomischen Gestaltung von Benutzeroberflächen . 54
4.3.1 IDA 54
4.3.2 GUIDEAS 56
4.3.3 Interface-Ratgeber 57
4.3.4 GUIDE 57
4.3.5 Gegenüberstellung der Werkzeuge 58
4.4 Erkenntnisse aus der Werkzeuganalyse 59

5 Konzeption einer unterstützenden Komponente für den Porsche AG HMI-Entwicklungsprozess 61
5.1 Der HMI-Entwicklungsprozess der Porsche AG 61
5.2 Unterstützungsmöglichkeit in den Prozessphasen 65
5.2.1 Vorentwicklung 66
5.2.2 Serienentwicklung 66
5.2.3 Serienbetreuung 70
5.2.4 Projektübergreifende Unterstützung 70
5.3 Ableitung des Konzepts für die Beratungskomponente und das Wissensmanagementsystem 71
5.4 Konzept der Beratungskomponente 72
5.4.1 Formalisierbarkeit der Dimensionen 72
5.4.2 Integration des Konzepts in die Werkzeugkette 80
5.4.3 Architektur der Beratungskomponente 91
5.4.4 Erkenntnisse aus dem Konzept 92
5.5 Konzept des Wissensmanagementsystems 94
5.5.1 Systemarchitektur 95
5.5.2 Integration des Konzepts in den Entwicklungsprozess 98

5.5.3 Zugriff auf das Wissensmanagementsystem . . . 101
5.5.4 Anwendungsbeispiel . . . 102
5.5.5 Erkenntnisse aus dem Konzept . . . 105
5.6 Mehrwert der Unterstützungskomponenten . . . 106
5.6.1 Prüfung der Kriterien nach DATech . . . 106
5.6.2 Erkenntnisse aus der Bewertung . . . 112

6 Fazit und Ausblick **115**

Verzeichnisse **120**
Abbildungsverzeichnis . . . 121
Abkürzungsverzeichnis . . . 121
Literaturverzeichnis . . . 123

Anhang **131**

A Fragebogen **131**

B Werkzeuge zur Unterstützung von Usability-Entwicklungsprozessen **137**

C Beispiel XAML **139**

D MetaEdit+ Report **143**

Kapitel 1

Einleitung

1.1 Ausgangssituation

Bei der Entwicklung interaktiver Systeme im Bereich der Automobilentwicklung spielt die gebrauchstaugliche und ergonomische Gestaltung von Benutzerschnittstellen eine zunehmend wichtige Rolle (Burmester et al. 2005). Im Gegensatz zur Entwicklung von Mensch-Maschine-Schnittstellen für Geräte wie Mobiltelefone, Fernsehgeräte oder andere moderne Haushaltsgeräte, müssen bei der Entwicklung von Fahrerinformationssystemen besondere Rahmenbedingungen beachtet werden. Es ist nach wie vor ein Trend bei vielen Autoherstellern immer mehr Funktionen in ihre Fahrzeuge zu integrieren (Infotainment, Internet, Informations- und Telematikdienste). Der Funktionsumfang nimmt unbeirrt zu und bereits etablierte Funktionen werden laufend erweitert. Dabei ergeben sich viele Abhängigkeiten, wodurch die Entwicklung eines Gesamtsystems immer komplexer wird (Von der Beek et al. 2001). Parallel dazu ist es außerdem erforderlich, die Schnittstelle zwischen Mensch und Technik an diese erhöhte Komplexität anzupassen.

Die Bedienung eines Fahrerinformationssystems geschieht meist während der Fahrt. Die jeweiligen Funktionen, wie zum Beispiel *Navigation* oder *Radio* müssen dementsprechend auch während der Fahrt bedienbar sein, ohne dass die Interaktion mit dem System den Fahrzeugführer zu sehr von der eigentlichen Fahraufgabe ablenkt. Das bedeutet jedoch, dass der Fahrer nicht seine volle Aufmerksamkeit der Interaktion mit dem System widmen kann. Diese Gegebenheit gilt es bei der Entwicklung zu beachten. Eine zu große Ablenkung vom Steuern des Fahrzeugs würde eine enorme Gefahrenquelle für den Fahrer und andere Verkehrsteilnehmer bedeuten. Aus diesem Grund wird die Bedienung eines Fahrerinformationssystems als Sekundäraufgabe bezeichnet, das Steuern des Fahrzeugs dementsprechend als Primäraufgabe. Diese Tatsache hat einen großen Einfluss auf die Entwicklung von Fahrerinformationssystemen und muss im Besonderen bei der Entwicklung von Interaktionskonzepten und Benutzerschnittstellen berücksichtigt werden. Des Weiteren können durch gebrauchstaugliche Benutzerschnittstellen Fehlbedienungen vermieden werden, welche im Fahrzeug ebenfalls von sicherheitskritischer Bedeutung sind. Im Gegensatz zur Entwicklung anderer interaktiver Systeme haben die Entwickler bei der Gestaltung eines Fahrerinformationssystems geringere Freiheiten. Dies ist einerseits auf eine strenge gesetzliche Vorgabe

zurückzuführen, durch die unter anderem die zu verwendende Symbolik und Anordnung (vgl. ISO 15005) geregelt ist. Andererseits schränkt die Hardwareplattform die Entwickler zusätzlich in ihren Möglichkeiten ein. Das hängt beispielsweise mit den standardisierten Bussystemen MOST[1] und CAN[2] genauso wie mit den direkt verwendeten Komponenten des Fahrerinformationssystems (Bildschirm, Bedienelemente etc.) zusammen.

Ein weiterer maßgeblich zu beachtender Punkt ist das arbeitsteilige Vorgehen zwischen Automobilherstellern und Zulieferern. In der Automobilindustrie ist dieses Verhältnis besonders stark ausgeprägt. Die Anforderungen eines Systems, wie zum Beispiel die Definition einzelner Funktionen, deren Interaktionskonzepte sowie die Benutzerschnittstellen, werden meist auf Seiten der Automobilhersteller erarbeitet. Die Umsetzung hingegen erfolgt größtenteils durch die Zulieferer. Die Hersteller sind wiederum für die Abnahme, Überprüfung und Integration in das Gesamtsystem *Fahrzeug* verantwortlich. Durch diese enge Verzahnung der Aufgaben ist es sehr wichtig, dass vorab klare Schnittstellen und Kompetenzbereiche zwischen den Entwicklungspartnern vereinbart werden (Von der Beek et al. 2001). Diese Tatsache hat bei der Definition eines Entwicklungsprozesses eine besondere Relevanz. Aus den genannten Gründen ist es eine große Herausforderung, ein einfaches und konsistentes Bedienkonzept für ein Fahrerinformationssystem zu entwickeln. Welche Bedeutung dabei der Gebrauchstauglichkeit zukommt, zeigt sich besonders hinsichtlich der Verbreitung der Systeme in den Fahrzeugen. Im Jahr 2006 konnte ein deutliches Plus der Neuzulassungen von Fahrzeugen mit Fahrerinformationssystemen gegenüber 2005 erzielt werden (Geländewagen +16,5 %, obere Mittelklasse +1,5% teure Oberklasse +10,6 %)[3]. Automotive Unternehmen können die gebrauchstaugliche und ergonomische Gestaltung der Produkte dadurch verstärkt als Verkaufsargument einsetzten.

1.2 Motivation

Um die Bedienbarkeit der Systeme zu gewährleisten und den Benutzer in den Vordergrund zu stellen, ist es sinnvoll während des gesamten Entwicklungsprozesses von Mensch-Maschine-Systemen kontinuierlich Evaluationen durchzuführen. Dabei ist es von Vorteil bereits in frühen Entwicklungsphasen Usability-Tests durchzuführen. Mit den Ergebnissen können Bedienkonzepte und Gestaltungsrichtlinien sehr genau überprüft werden und eventuelle Fehlentwicklungen frühzeitig erkannt und beseitigt werden. Spätere Korrekturen sind mit deutlich höherem Aufwand verbunden. So lässt sich nach Mayhew (1999) durch sehr frühe Tests die Entwicklungszeit um bis zu 40% verkürzen. Nach einer Studie des Fraunhoferinstitutes, welche von Freise (2003) durchgeführt wurde, können die Kosten für eventuelle Nachbesserungen am fertigen Produkt um das Zehntausendfache höher sein als zu Beginn eines HMI[4]-Entwicklungsprozesses.

[1] Media Oriented Systems Transport: Es handelt sich dabei um ein serielles Bussystem zur Übertragung von Audio- und Video-, Sprach- und Datensignalen über Lichtwellenleiter

[2] Controller Area Network: Es handelt sich dabei um ein asynchrones, serielles Bussystem, das von Bosch für die Vernetzung von Steuergeräten in Automobilen entwickelt und 1985 zusammen mit Intel vorgestellt wurde.

[3] Studie des Kraftfahrtbundesamtes, 2006

[4] Human-Machine Interface, deutsch Mensch-Maschine Schnittstelle

Um dies zu verhindern, empfehlen sich aus Erfahrung bestimmte Methoden des Usability-Engineering (Fokusgruppen, Strukturlegen, strukturiertes Interview, heuristische/empirische Evaluation etc.), die in einem strukturierten Entwicklungsprozess die nutzerorientierte Entwicklung von Bediensystemen ermöglichen. Neben geeigneten Methoden werden für Usability-Tests leistungsfähige Prototypen benötigt, die sowohl mit angemessenem Aufwand erstellt werden können, als auch eine realistische bzw. zielführende Bewertung der umgesetzten Bedienkonzepte und Gestaltungsrichtlinien erlauben. Da im Usability-Engineering Spezialisten aus unterschiedlichen Disziplinen (Entwickler, Psychologen, Designer, Usability Engineer etc.) zusammenarbeiten, können anschauliche Prototypen nicht nur für die Evaluation mit Probanden eingesetzt werden. Darüber hinaus können sie wesentlich zur Verbesserung der Kommunikation zwischen den am Entwicklungsprozess beteiligten Personen beitragen. Damit bereits im Entstehungsprozess ein aktiver Austausch zwischen den Beteiligten ermöglicht und gefördert wird, werden zur Erstellung von Prototypen leistungsfähige Konzepte benötigt. Bereits in frühen Projektphasen sind Erprobungen anhand erlebbarer virtueller Prototypen aus oben genannten Gründen sehr wichtig. Dafür müssen konventionelle Spezifikationsinstrumente weiterentwickelt werden, um eine effiziente Erstellung derer zu ermöglichen.
Helfen kann hierbei neben den Konzepten des Usability-Engineering auch der Einsatz etablierter Methoden des Software Engineering. Zu den Letztgenannten zählt insbesondere die modellgetriebene Softwareentwicklung einschließlich Codegenerierung. Einen viel versprechenden Ansatz stellen formale bzw. semiformale Spezifikationen dar, die erhebliche Produktivitäts- und Qualitätspotenziale für den gesamten Entwicklungsprozess eröffnen können. Modellbasierte Spezifikationsverfahren werden für die Abbildung und Vernetzungen logischer Funktionen immer häufiger eingesetzt. Der Einsatz dieser Methoden bildet eine solide Grundlage für die Analyse, die Validierung und die Verifikation von Funktionsnetzen (Von der Beek et al. 2001). Die konsequente Anwendung und Anpassung von Notationen, Techniken und Methoden ist vor allem in den Bereichen Requirements-, Usability- und Software-Engineering notwendig. Um diese Ansätze in einem HMI-Entwicklungsprozess einsetzen zu können, ist es erforderlich, die bestehenden Schwachstellen konventioneller Prozesse und Methoden aufzudecken und zu überwinden. Die sich ständig wandelnden Bedingungen, bei der Gestaltung von ergonomischen Benutzerschnittstellen führen zu veränderten Anforderungen an die Entwickler (Oed et al. 2001). Deshalb wird im Rahmen des Usability Engineering gefordert, dass im gesamten Systementwicklungsprozess die ergonomischen Aspekte beachtet werden sollen. Darüber hinaus nimmt mit stetig wachsender Systemfunktionalität der Grad der Standardisierung zu.
Ferner gibt es für die interaktiven Systeme diverse Guidelines[5] und Normen, die aufgrund der genannten Aspekte immer umfangreicher werden. Infolgedessen weiten sich die Anforderungen an einen Entwickler ständig aus, weil sie bei der Systementwicklung berücksichtigt werden müssen. Erschwerend kommt hinzu, dass die Normen größtenteils sehr allgemeingültig formuliert sind.

[5]Guidelines sind Richtlinie für die Gestaltung von Benutzerschnittstellen.

Dieser Problematik wird dadurch begegnet, dass den Entwicklern gewisse Freiheiten im Umgang mit Normen und Guidelines eingeräumt werden. Manchen Entwicklern fehlt allerdings die Orientierung im Umgang mit Normen und Standards. Dies kann sogar soweit führen, dass die Normen, Standards und Guidelines nicht angewendet werden. Nach einer Studie von Souzan und Bevan (1990) lag die Fehlerrate von Entwicklern bei der Systemgestaltung mithilfe Guidelines bei 91%. Blume et al. (2005) führen die hohe Fehlerrate auf die unterschiedlichen Sichtweisen sowie auf die persönliche Erfahrung zurück. Darüber hinaus fehlen den Entwicklungswerkzeugen von heute gänzlich die Möglichkeiten, die Guidelines und die vorherigen Erfahrungen aus den Projekten aufzunehmen.
Es befinden sich über 130 Werkzeuge auf dem Markt, die Programmierer bei der Gestaltung von Bediensystemen unterstützen (Oed et al. 2001). Keines der Werkzeuge vermittelt den Entwicklern das nötige Wissen in den Phasen des Usability-Engineerings (Oed et al. 2001). So sollten sich die Systeme neben dem reinen Softwareentwurf auch mit den Bereichen Beratung und Wissensvermittlung befassen.
Dass dem Wissensmanagement ein sehr hoher Stellenwert in der Softwareentwicklung zukommt, konnten schon Wandtke et al. (1999) 1998 bei einer Untersuchung in den Unternehmen feststellen. Diese Tatsachen intensivieren die Problematik bei der Systementwicklung. Deshalb benötigen Entwickler bereits in frühen Phasen Unterstützung, um die Gebrauchstauglichkeit interaktiver Systeme zu gewährleisten.

1.3 Einordnung und Abgrenzung der Arbeit

Die vorliegende Masterthesis entstand im Kontext des Projektes MORE (**MO**del-based **RE**quirements engineering for automotive information systems).
Ziel des Projektes ist der Entwurf und die prototypische Umsetzung einer unterstützenden Werkzeugkette für den HMI-Entwicklungsprozess der Porsche AG. Dies geschieht im Rahmen einer Kooperation zwischen der Dr. Ing. h. c. F. Porsche AG[6] und der Arbeitsgruppe Mensch-Computer-Interaktion der Universität Konstanz. Im Rahmen des Projektes soll das aktuelle Vorgehen der Entwicklung von Mensch-Maschine-Schnittstellen bei der Porsche AG beobachtet und analysiert werden.
Auf Basis dieser Ergebnisse werden Verbesserungspotenziale herausgearbeitet und als Grundlage für den Entwurf einer entwicklungsbegleitenden Werkzeugkette heran gezogen. Die Werkzeugkette bietet eine automatische Erstellung von Prototypen an. Heilig (2007) betrachtet in seiner parallel zur vorliegenden Arbeit entstandenen Arbeit die Werkzeugkette als Beispiel für modellgetriebene Entwicklungsmethoden in einem Usability-Entwicklungsprozess. Die Werkzeugkette wurde von Heilig (2007) für das Fahrerinformationssystem der Porsche AG entwickelt. Beim PCM 2.1, dem Porsche Communication Management, handelt es sich um ein von Porsche entwickeltes Fahrerinformationssystem (vgl. Abbildung 1.1).

[6] Zur besseren Lesbarkeit wird im restlichen Teil der Arbeit Porsche AG verwendet.

Abbildung 1.1: Fahrerinformationssystem PCM 2.1 der Porsche AG

1.4 Zielsetzung der Arbeit

Ziel dieser Arbeit ist die Konzeption einer Komponente, die den Entwickler bei der Erstellung gebrauchstauglicher interaktiver Systeme unterstützen soll. Dabei dient die Werkzeugkette, welche in den Arbeiten von Bock (2007) und Heilig (2007) entstanden ist, als Grundlage. Darüber hinaus soll geprüft werden, wie die neue Komponente in den HMI-Entwicklungsprozess eingebettet werden kann. Im Anschluss soll der HMI-Entwicklungsprozess erneut betrachtet werden und die neue Situation mit der Unterstützungskomponente evaluiert werden.
Um ein Unterstützungswerkzeug zu konzipieren, das noch weitere Aufgabenbereiche im Entwicklungsprozess abdecken kann, sollen zuerst die Nutzerwünsche der Entwickler in Form eines Fragebogens und Interviews aufgenommen werden. Die Evaluation des HMI-Entwicklungsprozesses der Porsche AG soll zeigen, wo noch verstärkt Unterstützungsbedarf besteht. Aus der umfangreichen Analyse des HMI-Entwicklungsprozesses wird anschließend ein Konzept abgeleitet, welches die Entwickler während der Gestaltung ergonomischer Benutzeroberflächen unterstützen soll. Dabei soll die neue Komponente, die den Entwickler in den Bereichen *Oberflächen-*, *Informations-* und *Interaktionssdesign* unterstützt, wenn möglich in die Werkzeugkette integriert werden.
Zum einem wird den Entwicklern eine *Beratungskomponente* für die Spezifikation ergonomischer Bediensysteme bereitgestellt. Beispielsweise soll so der Entwickler bei der Einhaltung von ergonomischen Standards angeleitet werden. Darüber hinaus sollen dem Entwickler vorgefertigte ergonomische Dialogbausteine wie beispielsweise PCM-Bildschirme bereitgestellt werden. Die Prüfung während der Entwicklung interaktiver Systeme soll automatisiert erfolgen.
Zum anderen wird ein *Wissensmanagementsystem* zur Verfügung gestellt. In diesem Hilfesystem können die Entwickler Wissen bereitstellen und auch Wissen bei Bedarf abrufen. Ferner soll durch das Informationssystem die Koordination und projektinterne Kommunikation gefördert werden. Der Import von Dokumenten wie beispielsweise Guidelines und Normen muss ohne großen Programmieraufwand gewährleistet sein, so dass sich die am Entwicklungsprozess beteiligten Personen auf ihre Domänen konzentrieren können.

1.5 Aufbau der Arbeit

Die vorliegende der Arbeit gliedert sich in 6 Kapitel. Die zugrunde liegende Ausgangssituation, Motivation und die Zielsetzung wurden im 1. Kapitel vorgestellt.
Im 2. Kapitel wird der aktuelle HMI-Entwicklungsprozess der Porsche AG im Rahmen einer Befragung eingehend untersucht. Anschließend werden aus den Ergebnissen der Untersuchung die Anforderungen für eine Unterstützungskomponente abgeleitet.
Im 3. Kapitel werden wissenschaftlichen Grundlagen, die im Kontext der Arbeiter relevant sind, eingehend vorgestellt. Dazu werden zuerst die bekanntesten Usability-Engineering-Prozesse analysiert. Zusätzlich werden Qualitätsmerkmale der Entwicklungsprozesse betrachtet. Des Weiteren werden Möglichkeiten präsentiert, inwiefern Hilfesysteme derzeit in herkömmlichen Softwareanwendungen zur Verfügung gestellt werden. Ferner werden die grundlegenden Begriffe erläutert, die im Zusammenhang mit Wissensmanagement stehen. So bietet das Kapitel einen Überblick über die grundlegenden Konzepte und Theorien im Bereich des Wissensmanagements.
Unterstützungsmöglichkeiten zur Gestaltung ergonomischer Benutzerschnittstellen für Usability-Entwicklungsprozesse werden im 4. Kapitel vorgestellt. Die eingehende Analyse der Werkzeuge bildet die Basis für die Unterstützungskomponente der Porsche AG.
Im 5. Kapitel wird das abgeleitete Konzept der Beratungs- und Wissenskomponente thematisiert. Ferner wird untersucht, in welchen Phasen die Beratungskomponente sinnvoll eingesetzt werden kann, um die Benutzer auch zweckmäßig zu unterstützen. Den Abschluss bildet die Überprüfung der Qualität des HMI-Entwicklungsprozesses der Porsche AG mit der neuen Unterstützungskomponente anhand des DATech Prüfhandbuches.
In Kapitel 6 werden die Erkenntnisse und Potenziale des vorgestellten Konzepts abschließend zusammengefasst und ein Ausblick gegeben.

Kapitel 2

Analyse des aktuellen Porsche AG HMI-Entwicklungsprozesses

Das folgende Kapitel verdeutlicht den aktuellen Entwicklungsstand des Porsche AG HMI-Entwicklungsprozesses. Dieser wurde anhand einer Evaluation mithilfe von Fragebogen und Interviews analysiert. Darüber hinaus wurde der Entwicklungsprozess drei Monate vor Ort beobachtet, um die Kommunikationswege im Prozess kennen zu lernen.
Der HMI-Entwicklungsprozess wird parallel in der Arbeit von Heilig (2007) mit dem DA-Tech Prüfhandbuch untersucht. Die ausgiebige Analyse dient demnach als Ausgangslage für weitere methodische Schritte innerhalb der Forschungskonzeption und muss im Folgenden als Ausgangsbasis betrachtet werden.

2.1 Zielsetzung der Analyse

Im Rahmen dieser Arbeit wurde zunächst der HMI-Entwicklungsprozess evaluiert. Dazu wurden die am Entwicklungsprozess beteiligten Personen in Form eines strukturierten Interviews, welches in der Forschung auch als standardisiertes Interview bekannt ist, befragt. Die Befragung erfolgte, um die potenziellen Nutzer der Werkzeugkette früh in die Entwicklung mit einzubeziehen. Die Einbeziehung der Entwickler ist integraler Bestandteil der Arbeit, um nicht das Produkt an diesen vorbei zu entwickeln. Durch eine bessere, den Bedürfnissen und Wünschen der Entwickler angepasste Systemfunktionalität und -gestaltung wird die Akzeptanz gesteigert. Dadurch steigt auch die Zufriedenheit der Endnutzer. Die Auswirkungen kommen sowohl den Entwicklern als auch dem Unternehmen zugute.
Ziel der Untersuchung war es, ein Verständnis für die Aufgaben und Tätigkeitsfelder der Personen zu bekommen. Darüber hinaus wurden die Personen auch über die aktuell verwendeten Werkzeuge befragt. Basierend auf den gewonnenen Ergebnissen der Befragten wurde ein Anforderungskatalog entwickelt, um eine rechnerbasierte Werkzeugunterstützung zu entwickeln.

2.2 Gang der Untersuchung

Die Personen wurden im Rahmen eines strukturierten Interviews anhand eines sogenannten Leitfadens befragt. Bei der Untersuchung wurde der Leitfaden in Form eines Fragebogens umgesetzt, wobei nicht die Person selbst den Fragebogen ausfüllte, sondern der Interviewer. Somit konnte der Interviewer bei Bedarf gezielt nachfragen. Der Vorteil dieser Befragungsart ist, dass *quantitative* (Auswertung Fragebogen) mit *qualitativen* Ergebnissen (Aussagen der Personen) kombiniert werden können. Der Interviewer hat so auch die Möglichkeit, bei eventuellen Fragen Rede und Antwort zu stehen. Darüber hinaus wurde ein Protokoll von einer dritten Person erstellt, um alle Details des Interviews in schriftlicher Form zur Verfügung zu haben.
Die Akquisition der Versuchspersonen erfolgte über ein gezieltes Mailing der in Frage kommenden Personen aus allen am Entwicklungsprozess beteiligten Personenkreisen (Abteilung Elektrik/Elektronik Vorentwicklung (EEV) und Elektrik/Elektronik Infotainment (EEI). Zwölf Personen nahmen insgesamt an der Befragung teil. Einschränkungen bezüglich des Mitarbeiterprofils gab es nicht.
Der fünfseitige Fragebogen (vgl. Anhang A) wurde in die Themengebiete *Aufgabenbereich*, *Beurteilung der Kriterien* und *bisherige Vorgehensweise* aufgeteilt. Der Fragebogen wurde in Form von Freitextfeldern und Ankreuzmöglichkeiten gestaltet. In dem folgenden Abschnitt wird erläutert, was genau in den einzelnen Themengebieten erfragt wurde.
Im ersten Teil des Fragebogens mussten die jeweiligen Personen Angaben zu ihrem Aufgabenbereich im HMI-Entwicklungsprozess machen. Darüber hinaus sollten sie ihre Tätigkeit den Bereichen *Design*, *Inhalt* oder *Bedienablauf* zuordnen. Hierbei war eine Mehrfachnennung möglich, denn es gab Personen, die für mehrere Aufgaben verantwortlich waren.
Während der Analyse des HMI-Entwicklungsprozesses wurden anhand der Anforderungen an die Werkzeugkette Kriterien abgeleitet. Die Kriterien ließen sich in die Dimensionen *Funktionsumfang*, *automatisierte Ausgabe* und *Technik* kategorisieren. Die am Entwicklungsprozess beteiligten Personen mussten die Kriterien in w*eniger wichtig*, *wichtig* und sehr *wichtig* einteilen. Konnte zu einem Punkt keine Aussage getroffen werden, wurde dieser einfach übersprungen. So haben beispielsweise einige der Befragten das Kriterium *Technik* übersprungen, weil sie zu diesem Punkt keine Aussage treffen konnten. Im folgenden werden die Kriterien des *Funktionsumfangs* vorgestellt.
Anpassbarkeit: Die Werkzeugkette musste für verschiedene Projekte des Hauses Porsche angepasst werden können. Beispielsweise sollte es möglich sein, ohne großen Aufwand unterschiedliche Projekte zu spezifizieren, beispielsweise PCM oder Kombiinstrument.
Modellierung Bedienabläufe: Die grafische Modellierung von Bedienkonzepten soll möglich sein. Darüber hinaus sollte eine mögliche Unterscheidung von Mikro[1]- und Makroverhalten[2] angedacht werden.
Vertraute Notation: Die Werkzeugkette sollte das Arbeiten mit vertrauten Notationen berücksichtigen, sodass sich die Personen voll auf ihre Domäne konzentrieren können. Zur Ablaufmodellierung werden beispielsweise dem *Usability-Engineer* illustrierte PCM-Bildschirme zur Verfügung gestellt.

[1] Detaillierte Spezifikation der internen Systemzustände.
[2] Illustrierten State Charts zur Spezifikation der sichtbaren Bildschirminhalten.

Usability: Die Werkzeugkette sollte intuitiv bedienbar sein, um eine effiziente wie effektive Arbeit zu ermöglichen.
Wiederverwendbarkeit: Einmal erstellte Vorlagen oder Templates sollten in verschiedene Projekte portierbar sein.
Automatische Prüfung: Das Werkzeug sollte die Personen auf mögliche Fehler bei der Entwicklung von Bediensystemen hinweisen. Die erstellten Modelle wurden automatisch auf selbst definierte Regeln geprüft. Vorstellbar wäre eine Prüfung der Rechtschreibung sowie die Länge der Menüeinträge bei der Entwicklung eines Fahrerinformationssystems. Ferner sollte auch die Konsistenz der Bedienabläufe auf mögliche *Deadlocks* geprüft werden. Das System informiert den User, sobald keine Rückkehr aus dem Navigationspfad möglich ist.
Gemeinsames Arbeiten: Die am Entwicklungsprozess beteiligten Personen arbeiten an einer gemeinsamen Datenbasis. Außerdem sollte durch eine Versionsverwaltung Transparenz in das Projekt kommen.
Rollenkonzept: Die Werkzeugkette sollte den Personen jeweils eine rollenspezifische Sicht anbieten. Der Administrator konnte die Zugriffsrechte für die Rollen vergeben.
Layoutmöglichkeit von Bildschirmen und Blenden: Die Werkzeugkette sollte beispielsweise für das Design von PCM-Bildschirmen eine Zeichenfunktion beinhalten. Die Personen durften nicht in ihrer Kreativität eingeschränkt werden.

Im Folgenden werden die Kriterien der *Automatisierten Ausgabe* vorgestellt.
Spezifikation: Ein standardisiertes Dateiformat für die Kooperation von Zulieferern sollte verwendet werden.
Export in Standardformate: Der Export der Standardformate von *Office-Produkten* sollte gewährleistet sein, um so beispielsweise ein Lastenheft für Zulieferer zu generieren.
Testautomatisierung: Das Werkzeug sollte die Testautomatisierung unterstützen, denkbar war hier ein Vergleich zwischen Spezifikation und Bediensystem.
Usability-Tests: Die Werkzeugkette sollte in frühen Phasen die entwicklungsbegleitende Evaluation unterstützen. Probandentests könnten von potenziellen Kunden so schon fast zu jedem Zeitpunkt durchgeführt werden.
Simulation/virtuelle Prototypen: Bedienabläufe könnten mithilfe von virtuellen Prototypen früh ohne aufwendige Programmierung spezifiziert werden.

Im Folgenden werden die Kriterien der *Technik* vorgestellt.
Prozesssicherheit: Um einen reibungslosen Einsatz der Werkzeugkette in den Entwicklungsprozess der Porsche AG zu erreichen, soll ein hoher Reifegrad gewährleistet sein.
Einsatz von Standards: In der Werkzeugkette sollten standardisierte Technologien zum Einsatz kommen, um die oben genannte Prozesssicherheit zu garantieren.
Support: Der Support sollte für die Komponente der Werkzeugkette zuverlässig verfügbar sein, um so bei eventuellen Ungereimtheiten einen Ansprechpartner zu haben.
Schnittstellen zu anderen Werkzeugen: Das Werkzeug sollte eine API[3] für andere Programme bereitstellen. So könnten beispielsweise Daten aus der vorhanden *Doors-Datenbank*[4]

[3]Eine API (application programming interface), ist eine Schnittstelle, die von einem Softwaresystem anderen Programmen zur Anbindung an das System zur Verfügung gestellt wird.
[4]Programm für das Anforderungsmanagement.

integriert werden.
Darüber hinaus mussten die Befragten Angaben zur *bisherigen Vorgehensweise* machen. Ziel dieser Fragen war es, die bisherige Arbeitsweise sowie die verwendeten Werkzeuge der am HMI-Entwicklungsprozess beteiligten Personen zu analysieren. Die Mitarbeiter der befragten Abteilungen wurden gebeten, die aktuelle Vorgehensweise im Entwicklungsprozess zu beurteilen. Sie konnten Angaben zur aktuellen Vorgehensweise in den Kategorien *gut*, *neutral* oder *verbesserungsfähig* vornehmen. Zusätzlich befand sich auf dem Fragebogen ein Freitextfeld, dort konnte der Interviewer die Verbesserungsvorschläge eintragen. Ferner mussten Prozessbeteiligten Angaben machen, welche *Werkzeuge* sie im Entwicklungsprozess verwenden. Zusätzlich wurden der Einsatzzweck und der Anwendungsbereich erfragt. Die Angaben wurden ebenfalls vom Interview in das Freitextfeld übernommen. In der letzen Frage des Interviews wurden die Personen darüber befragt, welche *Funktionen* ihre tägliche Arbeit im Entwicklungsprozess erleichtern würden. Hierbei konnten sie spezielle Funktionen erwähnen, die in die Werkzeugkette integriert werden müssen.

2.3 Ergebnisse der Untersuchung

Bei der Auswertung der Befragung haben sich die Vorüberlegungen bestätigt: Es beteiligen sich sechs Rollen am Entwicklungsprozess, um der Komplexität der Produktentwicklung gerecht zu werden: *Produktmanager*, *Einkäufer*, *Designer*, *technische Experten*, *Usability-Engineers* und *Programmierer*. Folgende Abbildung 2.1 soll die Beziehung zwischen den Personen am HMI-Entwicklungsprozess aufzeigen.

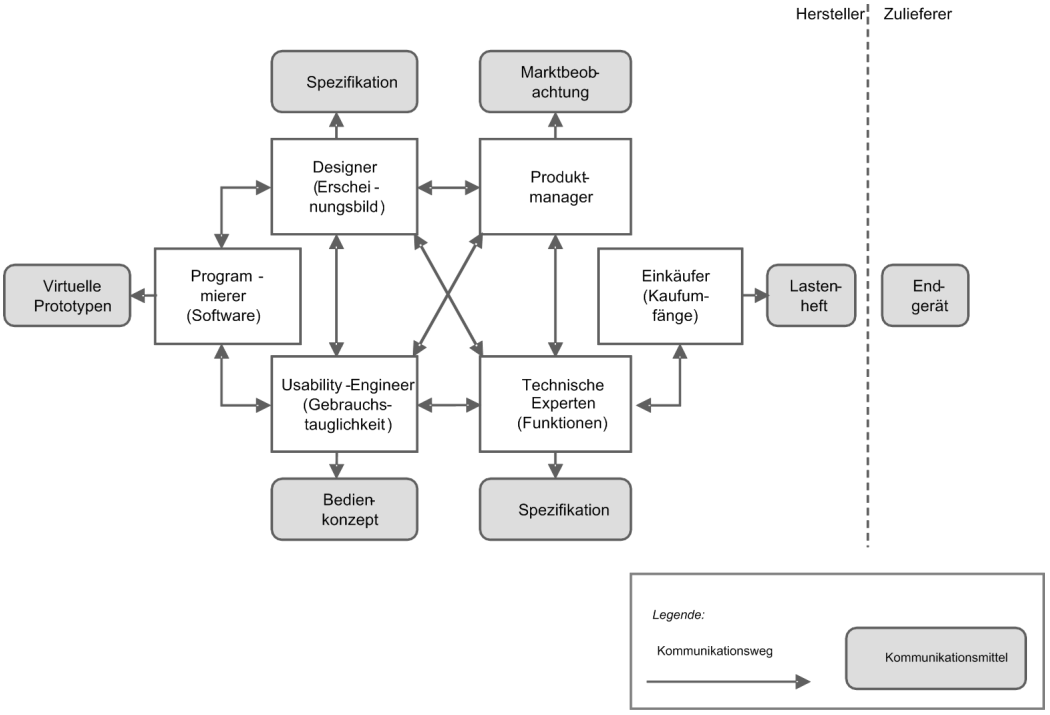

Abbildung 2.1: Am Entwicklungsprozess beteiligte Personen

Darüber hinaus wurde bei der dreimonatigen intensiven Betrachtung des Entwicklungsprozesses deutlich, wie die beteiligten Rollen miteinander kommunizieren (vgl. Abbildung 2.1).

Die Abbildung veranschaulicht auch Kommunikationsmittel der Rollen im Prozess.
Das *Produktmanagement* definiert strategische Zielvorgaben aus den Ergebnissen von Marktstudien und Kundenkliniken[5]. Es werden Anforderungen an das zu entwickelnde System bestimmt, die dem aktuellen Stand der Technik sowie den zukunftsweisenden Benutzeranforderungen gerecht werden sollen (Bock 2007). Die *Einkäufer* stellen primär die Schnittstelle zu den Entwicklungspartnern und den Herstellern sowie den Zulieferern dar. Als Kommunikationsmittel zwischen den *Einkäufern* und Herstellern wird das Lastenheft herangezogen.
Die *Designer* befassen sich mit der Oberflächengestaltung eines Bediensystems. Bei Fahrerinformationssystemen sind diese beispielsweise für das Bildschirm- und Blendendesign verantwortlich. Darüber hinaus gehört das Einhalten des *Corporate Designs* wie Formen, Farben, Schriftarten usw. zu ihrem Aufgabengebiet. Der *Designer* kommuniziert mit den anderen Rollen im HMI-Entwicklungsprozess, in dem er die Spezifikation des erarbeiteten Designs weiter gibt. Dies kann beispielsweise durch Grafiken geschehen.
Die *technischen Experten* arbeiten eng mit den *Usability-Engineers* und *Produktmanagern* zusammen. Sie prüfen und bewerten die Anforderungen, um die geforderte Funktionalität des Bediensystems zu gewährleisten. Darüber hinaus haben sie die verschiedenen Fahrzeugklassen und Ausstattungsvarianten immer im Blick. Als Kommunikationsmittel zieht der *technische Experte* die Spezifikationen der Anforderungen heran.
Die *Usability-Engineers* befassen sich mit der Bedienbarkeit des Systems. Ihre Zielvorgabe ist es, ein selbstbeschreibendes, erwartungskonformes sowie fehlerrobustes Bediensystem zu entwickeln. Dazu zählen unter anderem die Definition der Menüstruktur und die Gewährleistung eines konsistenten und logischen Verhaltens des Bedienkonzeptes für den Kunden. Ferner ist die Einflussnahme auf verschiedene Schriften- oder Farbparameter ebenso von Bedeutung. Diese Punkte werden jedoch eng mit dem *Designer* abgestimmt. Als Grundlage für die Kommunikation im Prozess benutzt der *Usability-Engineer* ausgearbeitete Bedienkonzepte in Form von *PowerPoint*-Folien oder *Excel*-Tabellen.
Der *Programmierer* setzt die geforderte Funktionalität für die Simulation um. Dies geschieht auf Basis der Anforderungen der im HMI-Entwicklungsprozess beteiligten Personen. Der *Programmierer* kommuniziert mit den Personen im Entwicklungsprozess, in dem er virtuelle Prototypen zur Verfügung stellt.

[5]Mithilfe von Kundenkliniken werden die aktuellen Produkte welche sich derzeit von Porsche auf dem Markt befinden getestet. So werden beispielsweise durch Benutzungshäufigkeitsanalysen die Funktionen im Fahrzeug daraufhin geprüft, wie sie vom Kunden angenommen werden.

Bei der Auswertung der Kriterien wurde der Anforderungskatalog zusätzlich zu den schon vorhanden 18 Kriterien priorisiert. Dazu wurden die Anforderungen in die Prioritätsklassen *Eins*, *Zwei* und *Drei* eingeteilt (vgl. Abbildung 2.2). Diese Einteilung sowie die Gewichtung der Kriterien erfolgte in einer Expertenrunde. Anforderungen, welche von den Befragten mit 80% (gewichtet: $> 9{,}6$), als *wichtig* oder *sehr wichtig* eingestuft wurden, erhielten die höchste Priorität. Der Prioritätsklasse *Zwei* wurden jene Anforderungen zugeordnet, die von 60% der Interviewpartner (gewichtet: >7,2) als (sehr) wichtig angesehen wurden. Die restlichen Anforderungen erhielten die geringste Priorität.

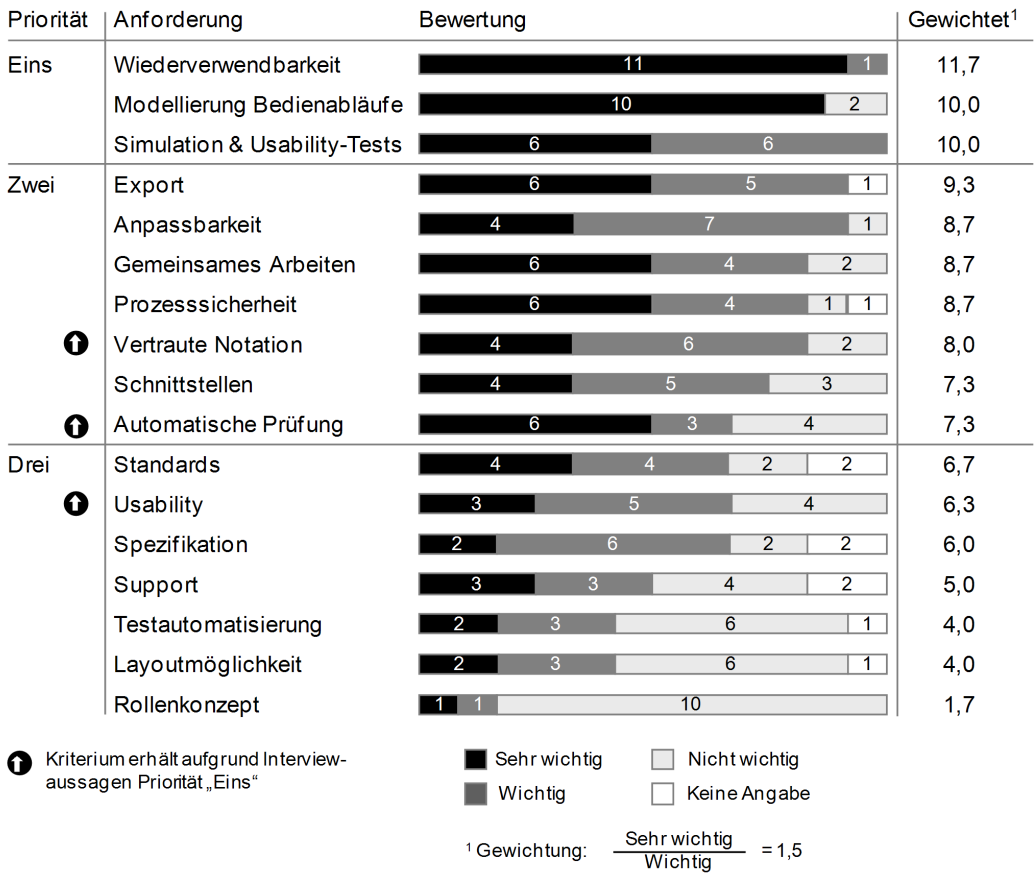

Abbildung 2.2: Auswertung der Kategorien mit priorisierte Rangfolge in Anlehnung an Bock (2007)

Die Auswertung konnte als Grundlage für die Entwicklung einer Werkzeugkette herangezogen werden. Jedoch konnten die Befragten beispielsweise mit den Begriffen *vertraute Notation* und *Usability* nicht umgehen, wenn man die Auswertung der Anforderungen mit den Interviewaussagen vergleicht. Aufgrund folgender Aussage mehrerer Probanden sind die beiden Anforderungen mit der Priorität *Eins* versehen worden. *”Die vertraute Notation und die Usability sind mir nicht wichtig, ich gehe aber dennoch davon aus, dass das Werkzeug so ähnlich aussieht, wie PowerPoint.”*
Die Entwickler waren sehr stark auf ihre gewohnten Büroanwendungen, wie beispielsweise MS-Office fixiert. Die angepasste Priorisierung der Kriterien *vertraute Notation* und *Usability* sollte somit Lösungen ausschließen, die aufgrund ihrer Gestaltung und Bedienbarkeit den Anschein erweckten, Spezialwerkzeuge zu sein, die nur mit erheblichem Einarbeitungsauf-

wand und unter Verzicht auf vertraute Darstellungs- und Interaktionsweisen genutzt werden können. Die Anforderungen *Simulation* und *Usability-Tests* wurden im Anschluss zusammengefasst, da die Befragten keinen Unterschied der Begrifflichkeiten feststellen konnten. Die Aussage: "*Ich kann doch mithilfe der Simulation, die Usability-Tests durchführen.*" kam von acht der zwölf Befragten. Sie gingen davon aus, dass die virtuellen Prototypen im Rahmen der entwicklungsbegleitenden Evaluation genutzt werden könnten. Abschließend ließen sich einige Anforderungen klar identifizieren. Zum einen wollten die Befragten ein Werkzeug, welches intuitiv bedient werden kann, aber dennoch im hohen Maße problemorientiert ist. Zum anderen wollten sie die spezifischen Bedienabläufe schnell im Rahmen von virtuellen Prototypen zur Verfügung gestellt bekommen. Darüber hinaus war es wichtig, dass die Spezifikationen in einem strukturierten Format vorliegen, das auch für Menschen lesbar ist. Neben dem internen Spezifikationsformat sollte auch ein herstellerübergreifendes Format zur Abstimmung mit den Zulieferbetrieben unterstützt werden. Einige große deutsche Automobilhersteller wie beispielsweise die AUDI AG und die BMW AG treiben dieses Format voran. Das Kriterium *Gemeinsames Arbeiten* erhielt von den Entwicklern auch einen hohen Stellenwert, denn derzeit ist dies nur unter schweren Bedienungen durchzuführen. Auf den Punkt *Gemeinsames Arbeiten* wird bei der Interviewauswertung genauer eingegangen, weil die Befragten im Interview die Aussagen genauer spezifizieren konnten.

Die Befragten waren mit der Vorgehensweise im HMI-Entwicklungsprozess nicht zufrieden (vgl. Abbildung 2.3). Sieben der Befragten fanden den Prozess derzeit *verbesserungsfähig*. Außerdem antworteten fünf Personen *neutral*. Auffällig ist, dass keine der befragten Personen die derzeitige Vorgehensweise im Entwicklungsprozess mit *gut* bewertete. Dieses Ergebnis zeigt, dass im jetzigen Prozess dringend Veränderungen vorgenommen werden müssen.

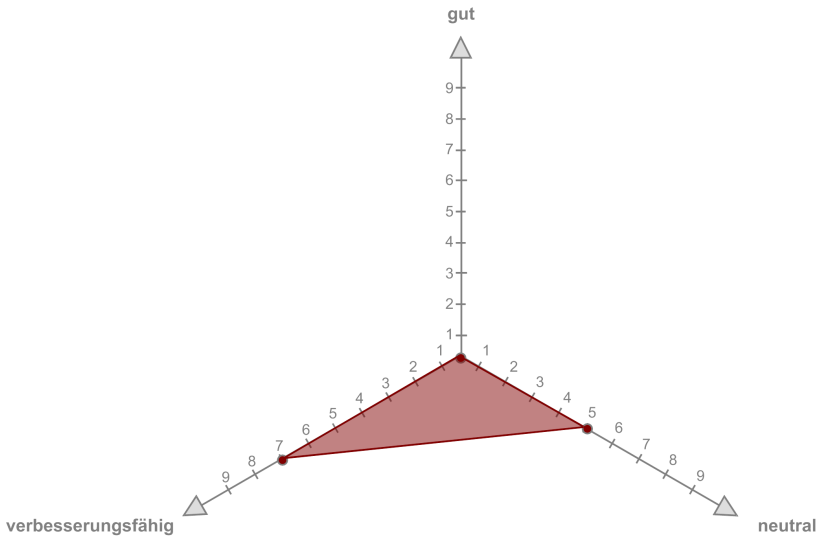

Abbildung 2.3: Bisherige Vorgehensweise

Die am häufigsten genannten Werkzeuge waren die klassischen Microsoft-Produkte, wie beispielsweise *PowerPoint* und *Word*. Es waren Mehrfachnennungen bei den benutzten Programmen erlaubt. Von den zwölf befragten Personen benutzten alle *PowerPoint* für die Spezifikation (vgl. Abbildung 2.4). Darüber hinaus verwendeten zehn Personen *Excel* und

sieben Personen *Word*. Neben den klassischen Büroanwendungen standen die Bildbearbeitungsprogramme wie beispielsweise *Photoshop* auf Platz vier. *Sonstige* Werkzeuge kamen nur bei bestimmten Personen zum Einsatz, die sie für Spezialanwendungen benötigen. Als Beispiel ist hier das Werkzeug *Doors* zu nennen, welches als Spezifikationswerkzeug für einen Zulieferer benutzt wird. Folgende Abbildung soll die Anzahl der eingesetzten Werkzeuge verdeutlichen.

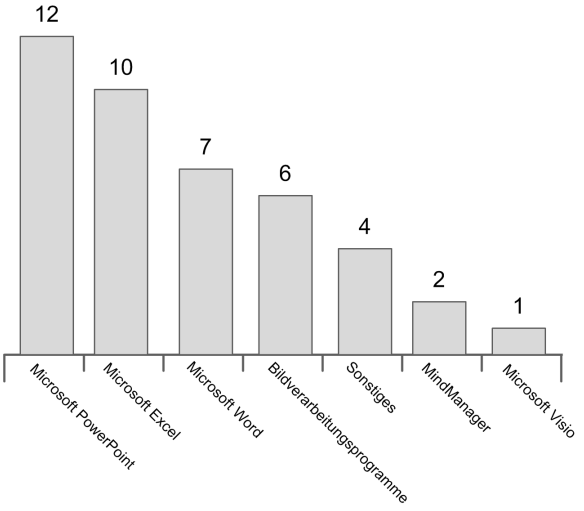

Abbildung 2.4: Häufigkeitsverteilung Spezifikationswerkzeuge

Die beteiligten Rollen benutzen zum größten Teil Produkte aus dem Hause Microsoft, wie *PowerPoint*, *Word* und *Excel*. Aus diesen Grund ist der Entwicklungsprozess der Porsche AG sehr stark papiergetrieben. So werden beispielsweise Spezifikationen untereinander in Papierform ausgetauscht. Bei der dreimonatigen Beobachtung ist auch klar geworden, dass zum Beispiel die *technischen Experten* nicht mit der gleichen Software ihre Funktionen spezifizieren. Zum einen werden die Funktionen in *Excel* und zum andern in *Word* spezifiziert. So ist der Austausch der Arbeitsergebnisse unter den *technischen Experten* mit einem erheblichen Mehraufwand verbunden. Die heterogene Werkzeuglandschaft wird in Abbildung 2.5 nochmals verdeutlicht. Zusätzlich werden die Werkzeuge in Verbindung mit den Rollen betrachtet.

	Microsoft PowerPoint	Microsoft Excel	Microsoft Word	Bild-bearbeitung	Sonstige	Mind-manger	Microsoft Visio
Produktmanagement	x	x	x				
Einkäufer	x	x	x				
Designer	x			x			x
TechnischeExperten	x	x	x		x		
Usability Engineer	x	x	x	x		x	x
Programmier	x	x	x		x		

Abbildung 2.5: Rollen und Werkzeuge im HMI-Entwicklungsprozess

Die gewünschten *Zusatzfunktionen* konnten von Entwicklern in einem Freitextfeld angegeben werden. Von den Funktionen, die zusätzlich den HMI-Entwicklungsprozess der Porsche AG unterstützen können, sind am häufigsten zum einem eine *Versionsverwaltung* der Daten (Do-

kumente, Grafiken von PCM-Bildschirmen usw.), zum anderen eine *zentrale Datenhaltung* in den Projekten genannt worden. Diese gaben neun der zwölf Befragten an. Eine repräsentative Aussage von einem Probanden lautete wie folgt: *"Ich muss mich ständig durch die verschiedensten Abteilungen im Bereich der Elektrik/Elektronik durchfragen, um die neusten Entscheidungen im Bereich Entwicklungsstand PCM 3.0*[6] *zu erfahren."*
Diese Aussage eines Entwicklers macht die Situation im HMI-Entwicklungsprozess der Porsche AG mehr als deutlich. Darüber hinaus wurde von zwei der zwölf befragten Personen eine Prüfung der Validität von Bedienabläufen gewünscht.

Bei den Aussagen der Befragten aus den Interviews wurde auch deutlich, dass sie mit dem Begriff *automatische Prüfung* der Benutzereingaben nicht zurecht kamen. Dies ist darauf zurückzuführen, dass sie im Umgang mit einem solchen System nicht vertraut waren. Nachdem der Interviewer den Begriff erläuterte, fanden neun der zwölf Befragten dieses Feature sehr wichtig, um eventuelle *Deadlocks* bei der Modellierung zu erkennen. Des Weiteren wurde nochmals bei den Interviews eine zentrale Datenhaltung im HMI-Entwicklungsprozess gefordert. Die Befragten haben hier bemängelt, dass sie in Sachen aktueller Stand von Entwicklung bzw. Design von Screens nicht auf dem Laufenden gehalten würden. So sagte ein Proband: *"Ich bastele mir meine eigenen Bildschirme und zieh diese dann näher ins neue Design, weil ich viel zu spät mitbekomme, wenn eine Veränderung des Designs spezifiziert wurde."*
Diese Aussage zeigt exemplarisch, dass Änderungen in aktuellen Prozessen nur unzureichend kommuniziert werden. Dies gilt beispielsweise für das Design und die Menüstruktur. Die oben genannten Punkte würden das *gemeinsame Arbeiten* an Bediensystemen erfordern. Es wurde deutlich, dass es sinnvoll wäre, die Entwicklungsstände transparent zu machen, d.h. die Befragten würden es für gut heißen, wenn es eine Historie der Entwicklung geben würde und die Entscheidungsstände transparent nachvollzogen werden können. Eine Änderung der Schriftfarbe von Menüpunkten wurde beispielsweise durchgeführt, weil bei einem Benutzertest im Fahrzeug herausgefunden wurde, dass die Schrift unter realen Bedingungen durch den geringen Kontrast nicht gelesen werden konnte.
Ferner wünschten sich drei der zwölf befragten Personen eine Art elektronisches Lexikon, um schnell auf Richtlinien im Rahmen des HMI-Entwicklungsprozesses zugreifen zu können. Bei genauerem Nachhaken des Interviewers auf die Frage, wo sie derzeit beispielsweise die Normen und Standards finden, kam als Antwort: *"Die hat nur der Usability-Engineer bei uns im Haus, wenn wir in diesem Zusammenhang etwas brauchen, ist dieser unser Ansprechpartner."* Dann weiter: *"Und außerdem ist es sehr mühselig, die richtigen Normen und Guidelines ausfindig zu machen".* So muss derzeit der *Usability-Engineer* mühselig die Richtlinien durchsuchen, um die Anfragen der anderen Personen zu befriedigen.

[6]Das PCM 3.0 Fahrerinformationssystem wird derzeit neu bei der Porsche AG entwickelt, es ist der Nachfolger vom PCM 2.1.

Die Aussagen aus den Interviews decken sich mit einer Untersuchung, welche von Oed et al. (2001) bei der DaimlerChrysler AG durchgeführt wurde. Thema der Untersuchung 2001 war: *"Welche Unterstützung wünschen Softwareentwickler beim Entwurf von Bedienoberflächen?"* Befragt wurden 16 Personen aus vier verschieden Entwicklerteams im Unternehmen, davon waren acht Softwareentwickler, vier Projektleiter und die Restlichen drei aus den Bereichen Konzeption, Produktdesign und Usability Testing. Die überwiegende Mehrheit (12 von 16) nannten unter den drei wichtigsten Unterstützungswünschen eine gründlichere Projektdokumentation sowie bessere Systeme zu Projektmanagement und -dokumentation (Oed et al. 2001). Ferner wollten über die Hälfte der Personen, Zugriff auf eine Art Erfahrungsdatenbank. Dieses Wissen könnte auch für Schulungszwecke und Weiterbildungsmaßnahmen der Mitarbeiter im Unternehmen herangezogen werden. Durch die Untersuchung von Oed et al. (2001) bei DaimlerChrysler bekräftigen sich die Anforderungen an eine Werkzeugunterstützung, um die Entwickler zu entlasten. Die Werkzeugunterstützung kann in Zukunft dabei helfen, interaktive Bediensysteme erfolgreicher zu entwickeln, da die Erfahrungen der Entwickler allen frei zugänglich gemacht werden.

2.4 Anforderungen an die Werkzeugunterstützung

Die Analyse ergab, dass die Befragten im Rahmen der Bediensystementwicklung Unterstützung benötigen, um die technische und organisatorische Komplexität der Produktentwicklung zu handhaben. Dazu wurde für den Porsche HMI-Entwicklungsprozess eine Werkzeugkette konzipiert, welche ausführlich in den Arbeiten von Bock (2007) und Heilig (2007) beschrieben wird. In Anlehnung an das *Seeheim-Modell* (Green 1986) und dem *Model-View-Controller* Entwurfsmuster (Balzert 1998) modularisiert Bock (2007) interaktive Bediensysteme für die Werkzeugkette in die Komponenten *Oberflächen-*, *Informations-* und *Interaktionsdesign* (vgl. Abbildung 2.6). Somit kann für die verschiedenen Dimensionen eine Werkzeugunterstützung implementiert werden und zusätzlich die Komplexität für die einzelnen Entwickler gesenkt werden (Bock 2007). Eine ähnliche Aufteilung der Dimensionen wird in vielen weitern Forschungsprojekten durchgeführt, exemplarisch ist hier MUSE II zu nennen (vgl. Kapitel 4). MUSE II unterscheidet in *zweckgebundene*, *interaktionsbezogene* und *präsentationsbezogene* Sicht.
Die Dimension *Informationsdesign* deckt die Aufbereitung und tatsächliche Darstellung der Informationsobjekte ab. Hier sind Fragestellungen zu lösen wie "*welche Funktionen werden angeboten?*" (Funktionsumfang) oder "*wie werden die Funktionen im System benannt?*" (Namensgebung). Das *Oberflächendesign* interaktiver Systeme behandelt Themen wie die Anordnung und Gestaltung der Oberflächenelemente auf dem Fahrerinformationssystem. Des Weiteren beinhaltet das *Oberflächendesign* die Farbgebung der sich auf dem Bildschirm befindenden Elemente.
Bedienabläufe und Menüstrukturen sind der Dimension *Interaktionsdesign* zuzuordnen. Im Folgenden wird der Aufbau der Werkzeugkette am Beispiel der Entwicklung eines Fahrerinformationssystems näher beschrieben. Dabei werden passende Anwendungen in die Werkzeugkette integriert und mit einem Quelltext-Generator und Software-Framework zu einer Werkzeugkette zusammengefügt (Heilig 2007). Das Hauptaugenmerk des *Interaktionsdesigns*

liegt darin, eine den Aufgaben angemessene, eine erwartungskonforme sowie eine konsistente Bedienung zu erreichen. Dazu zählt beispielsweise die Auswahl der Bedienelemente (Green 1986, Balzert 1998).

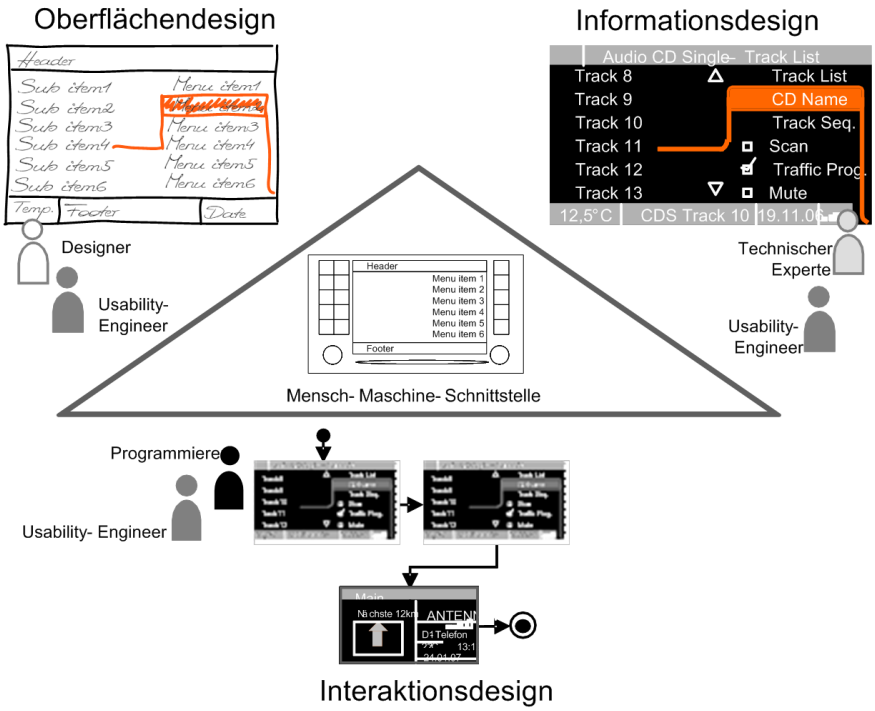

Abbildung 2.6: Aufbau der Werkzeugkette (Memmel et al. 2007)

Die Dimensionen *Informationsdesign* und *Interaktionsdesign* werden durch ein domänenspezifisches CASE-Tool unterstützt. Dazu wird eine eigene Modellierungssprache speziell für die Domäne entworfen. Mit dieser Sprache können die Informationsobjekte und Bedienstrukturen abgebildet werden. Im Gegensatz dazu wird für die Dimension *Oberflächendesign* ein GUI-Builder[7] verwendet (Heilig 2007). Zusätzlich zu den Werkzeugen für die Modellierung der drei Dimensionen kommt ein Quelltext-Generator und das Domäne-Framework hinzu. Der Quelltext-Generator extrahiert die Informationen der domänenspezifischen Modelle und bereitet diese für das Domäne-Framework auf. Er verbindet weiterhin die Informationen aus dem domänenspezifischen CASE-Tool mit der *Oberflächendesign*-Komponente und stellt die gewünschten Ausgaben, wie z. B. einen Prototypen des PCM 2.1, bereit (Heilig 2007).
Heilig (2007) stellt in seiner Arbeit einige Forschungsprojekte vor, bei denen modellgetriebenen Entwicklungsmethoden zum Einsatz kommen. Im Rahmen der vorliegenden Arbeit werden die Schwachstellen dieser Methoden aufgezeigt. Eine ausführliche Beschreibung der Werkzeuge sind in der Arbeit von Heilig (2007) zu finden. So zeigt sich, dass die bisherigen Forschungsprojekte beim Einsatz der modellgetriebenen Entwicklungsmethoden nur auf eine Phase im Entwicklungsprozess beschränkt sind.

[7]Softwarekomponente um grafische Benutzeroberfläche zu gestalten (engl. Graphical User Interface).

Abbildung 2.7 zeigt die untersuchten Projekte von Heilig (2007). Die Werkzeuge wurden in Entwicklungsphasen, in der sie hauptsächlich eingesetzt werden, der Komplexität der verwendeten Modelle sowie der Möglichkeiten der Weiterverarbeitung der Modellinhalte eingeteilt (Heilig 2007). Die Entwicklungsphasen werden ausführlich in Abschnitt 5.1 beschrieben.

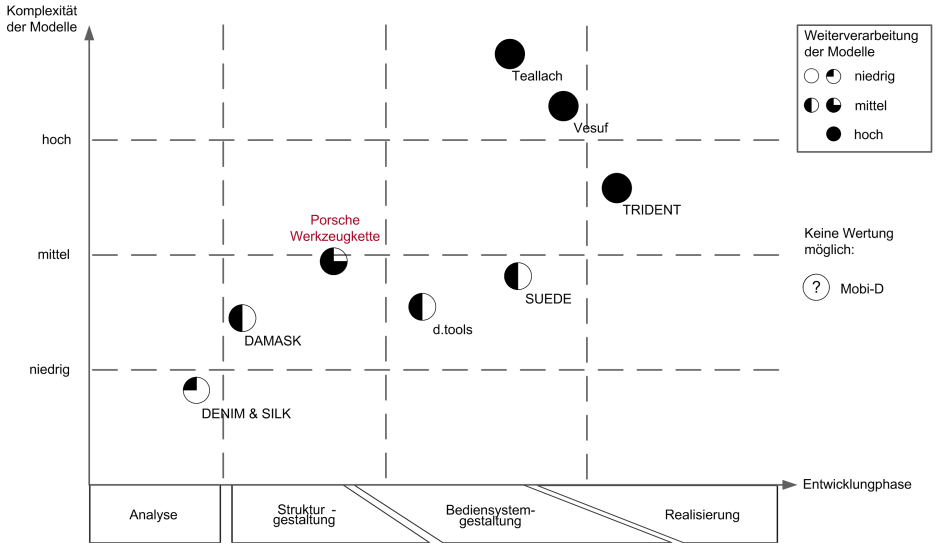

Abbildung 2.7: Klassifizierung der vorgestellten Forschungsprojekte (Heilig 2007)

Wie aus dem Schaubild zu entnehmen ist, gibt es für alle Phasen einer Entwicklung modellgetriebene Ansätze einer Werkzeugunterstützung. Aus dem Detaillierungsgrad der generierten Oberflächen und der Komplexität von den Modellen lässt sich ein Zusammenhang feststellen. Je komplexer die Modelle sind, desto detailreicher können die generierten Artefakte sein (Heilig 2007). Probleme bei Modellen treten häufig beim Festlegen des Abstraktionsniveaus auf, da es derzeit keine Standards für den Aufbau der Spezifikation der Modelle gibt. Darüber hinaus sind oft eine nicht ausreichende Ausdrucksstärke der Modellierungssprachen und ein hoher Lernaufwand als Nachteile zu verzeichnen (Heilig 2007).
Des Weiteren ist bei den vorgestellten Werkzeugen ein detailliertes *Oberflächendesign* schwer umzusetzen, da der Schwerpunkt auf der Modellierung des *Interaktionsdesigns* liegt. So wurde von Bock (2007) und Heilig (2007) ein Werkzeug konzipiert, das mit relativ einfachen Modellen möglichst detailreiche Oberflächenprototypen generieren kann. Somit können auch sogenannte Nicht-Experten, Bedienabläufe für interaktive System modellieren.

Durch den Ansatz von Heilig (2007) und Bock (2007) wurde gezeigt, dass eine durchgängige Werkzeugkette für die Unterstützung der *Serienentwicklung* im Prozess der Porsche AG einige Mehrwerte bietet, obwohl zunächst initialer Aufwand bei der Erzeugung der domänenspezifischen Sprache geleistet werden muss (Heilig 2007). Der große Vorteil der Werkzeugkette im Gegensatz zu anderen Forschungsprojekten aus dem modellbasierten Bereich ist, dass mit der Werkzeugkette derzeit schon ein detailliertes *Oberflächendesign* ermöglicht wird. Dies geschieht durch den GUI-Builder.
Zusätzlich muss jedoch der Entwickler bei den ständig wachsenden Anforderungen während des ganzen HMI-Entwicklungsprozesses, wie beispielsweise dem Anwachsen von Guidelines (ISO 15008) im Fahrzeug, unterstützt werden, so dass er sich besser auf seine eigentli-

che Entwicklungsdomäne konzentrieren kann. Das derzeitige Konzept der Werkzeugkette berücksichtigt jedoch ausschließlich die Phase der *Serienentwicklung*, die *Vorentwicklung* und *Serienbetreuung* werden vernachlässigt. Aus diesem Grund wird im Rahmen dieser Arbeit einen zusätzliche Unterstützungsfunktion entwickelt die den Entwickler während des ganzen HMI-Entwicklungsprozesses begleitet. Diese Unterstützungsfunktion basiert auf der entwickelten Werkzeugkette von Bock (2007) und Heilig (2007). Sie ist somit als zusätzliche Komponente der bisherigen Werkzeugkomponente zu betrachten. Darüber hinaus soll mit einer Unterstützungsfunktion im Rahmen des HMI-Entwicklungsprozesses der Abstimmungsaufwand der Personen stark reduziert werden, um bei eventuell auftretenden Fehlern frühzeitig reagieren zu können. Gleichzeitig resultiert daraus eine enorme Kosteneinsparung bei früher Fehlererkennung (Mayhew 1999). Die Berücksichtigung, den Entwickler schon in frühen Phasen des Entwicklungsprozesses zu unterstützen, muss integraler Bestandteil der gesamten Produktentwicklung sein. So können Qualitätsmerkmale bereits sehr für im Produktentstehungsprozess geprüft werden.
Die Befragung (vgl. Abschnitt 2.3) hat gezeigt, dass einige Entwickler eine Werkzeugkette mit integrierter Hilfskomponente befürworten. Darüber hinaus wurde das Potenzial im Bereich Produktmanagement deutlich gemacht. Die frühe Nutzerintegration zeigte, dass die Mitarbeiter sehr motiviert waren, den aktuellen Entwicklungsprozess zu verbessern. Man entschied sich bewusst für diese Methode, um die volle Unterstützung der Entwickler zu gewährleisten. So können die umfangreichen Erfahrungen der Entwickler in die Konzeption der Komponenten mit eingebracht werden.
Im folgendem Abschnitt liegt der Fokus auf der Definition der Anforderungsprofile für die Beratungskomponente und das Wissensmanagementsystem.

2.4.1 Anforderungsprofil der Beratungskomponente

Die Analyse der Benutzergruppen hat gezeigt, dass die Beratungskomponente ergonomische Gestaltungsrichtlinien vermitteln soll. Diese Richtlinien sollen bereits bei der Erstellung der Anforderungen des Produktes berücksichtigt werden.
Darüber hinaus sollten den Entwicklern vorgefertigte ergonomische Bausteine, beispielsweise Bildschirmdialoge bereitgestellt werden. Außerdem kann der Entwickler während des ganzen HMI-Entwicklungsprozesses anhand der Unterstützungskomponente die Qualität seiner Arbeit überprüfen lassen. Tritt ein Fehler auf, so wäre es hilfreich, wenn das System diesen erkennen und spezifizieren würde. Beispielsweise können durch eine Längenprüfung von Texten inkompatible Layoutspezifikationen vermieden werden. Erstellt ein Entwickler neue ergonomische Oberflächenkomponenten, so müssen diese in einer Art Bibliothek den Mitarbeitern frei zugänglich gemacht werden. Die Beratungskomponente soll mit diesen Modulen ineinandergreifen.

2.4.2 Anforderungsprofil der Wissenskomponente

Vor der Erstellung des Konzeptes der Wissenskomponente muss evaluiert werden, wie ein Wissenmangementsystem in den Bereich des Usability-Engineerings integriert werden kann. Dabei sollen Modelle aus dem Bereich des Wissensmanagements bewertet und aufgegriffen

werden. Ziel dabei ist eine Verbindung zwischen Usability-Engineering und dem Wissensmanagement zu schaffen, um so die Qualität des Porsche HMI-Entwicklungsprozesses zu steigern.
Für die Unterstützungskomponente sind folgende Funktionen angedacht: Die bisher erzielten Erfahrungen der Entwickler sollen strukturiert in der Datenbank abgelegt werden können (*Lessons Learned*, vgl. Abschnitt 5.1). Die *Lessons Learned* beinhalten bisher erworbenes Wissen aus vorangegangenen Projekten. Darüber hinaus sollen die Marktbeobachtungen der Produktentwickler transparent in der Wissensdatenbank vorliegen.
Durch Rollenkonzepte können die einzelnen Themen gegliedert und eingepflegt werden. Der Zugriff auf die Komponente muss ständig während des Entwicklungsprozesses möglich sein, um so die optimale Unterstützung zu ermöglichen. Beispielsweise muss der Entwickler jederzeit auf Normen, Standard und den Porsche Styleguide (Porsche 2000) zurückgreifen können. Nur so kann das Wissensmanagementsystem eine wertvolle Hilfestellung in der Entwicklung geben. Das Wissensmanagementsystem soll ferner als Kommunikationsinstrument im interdisziplinären Team dienen.
Umfangreiche Suchmöglichkeiten unter Einbeziehung von Metadaten werden durch gut strukturierte Inhalte ermöglicht. Weiter können neue Mitarbeiter durch innovative Gestaltungskonzepte mithilfe des Wissensmanagementsystems individuell geschult werden. Auch vorhandene Dokumente (pdf, doc, HTML etc.) müssen über eine Schnittstelle integrierbar sein.

Kapitel 3

Grundlagen

In diesem Kapitel werden die Grundlagen erläutert, die zum Verständnis der Ansätze und Verfahren notwendig sind, um das Konzept der Unterstützungskomponente zu begreifen. Zu Beginn wird dazu ein Überblick über die bekannten Usability-Engineering-Prozesse gegeben, um so die verschiedenen Methoden der Prozesse kennen zu lernen. Zusätzlich werden die Qualitätsmerkmale der Entwicklungsprozesse betrachtet, um so die Gebrauchstauglichkeit bereits sehr früh in der Produktentstehung zu gewährleisten. Im Anschluss soll in diesem Kapitel das grundlegende Verständnis für Hilfesysteme vermittelt werden. Dabei werden die verschiedenen bestehenden Ansätze charakterisiert. Ferner werden die grundlegenden Begriffe erläutert, die im Zusammenhang mit einem Wissensmanagement stehen. So bietet das Kapitel einen Überblick über die grundlegenden Konzepte und Theorien für das Wissensmanagement. Als weiterführende Literatur wird Nonaka und Takeuchi (1997), Davenport und Prusak (1998) und Probst et al. (2003) empfohlen.

3.1 Usability-Engineering-Prozesse

Ein besonders wirksamer Usability-Engineering-Entwicklungsprozess ist bei der Gestaltung von interaktiven Bediensystemen besonders wichtig, denn das Produkt soll effektiv, effizient und zufriedenstellend zu benutzen sein (Geis 2005). Um ein gebrauchstaugliches Produkt zu entwickeln, bedarf es eines sogenannten Lifecycles. Die Vorgehensmodelle intervenieren bereits vor der Phase des User-Interface-Designs. Ziel dabei ist es, für den Benutzter ein System zu entwerfen, mit dem eine effektive, effiziente und zufriedenstellende Bedienung möglich wird (Geis 2005). Das größte Hindernis für den Einsatz von Usability-Engineering ist, dass die Maßnahmen für den Software-Engineering-Prozess unabhängig von den Usability-Engineering-Maßnahmen geplant und durchlaufen werden (Metkzer und Reiterer 2002a). Keiner der im nächsten Abschnitt vorgestellten Entwicklungsprozesse kann ohne Anpassung für die Bediensystementwicklung herangezogen werden. Sie dienen lediglich als Vorgabe und müssen entsprechend auf die Rahmenbedingungen, welche das Produkt mit sich bringt, angepasst bzw. adaptiert werden (Woletz 2006).
Im Rahmen dieser Arbeit wird dem Leser nur eine kurze Einführung in den Usability-Engineering-Prozess gegeben, der versucht dieser Tendenz entgegen zu wirken. Detaillierte Ausführungen des Usability-Engineering-Prozess finden sich unter Burmester et al. (2005), Mayhew (1999), Beyer und Holtzblatt (1998) sowie Constantine und Lockwood (1999). In der

heutigen Zeit gibt es zahlreiche Ansätze solcher Vorgehensmodelle. Im Folgenden werden die Ansätze der Vorgehensmodelle kurz charakterisiert. Die bekanntesten Usability-Engineering-Vorgehensweisen werden in der Arbeit von Heilig (2007) ausführlich beschrieben.

3.1.1 Der Usability-Engineering-Lifecycle

Das Prozessmodell von Mayhew (1999) lässt sich in die Phasen *Requirements Analysis*, *Design/Testing/Development* und *Installation* unterteilen. Darüber hinaus kann der Usability-Engineering-Lifecycle mit drei Konzepten beschrieben werden.
Das Konzept des *User-Centred-Design* (benutzerzentrierte Gestaltung) charakterisiert die frühe Einbeziehung der Benutzer in den Prozess. Es dient der Steigerung der Produktqualität. Im zweiten Konzept dem *Usage-Centered-Design* steht die Erweiterung der Produktivität durch die Usability Optimierung im Vordergrund. So soll sich im Anschluss des Konzeptes ein Produkt entwickeln, welches weniger Fehler bei der Benutzung zulässt. Das zweite Konzept, das *Usage-Centered-Design* steht für die Erweiterung der Produktivität. Im letzten Konzept, dem Produkt-Styleguide, werden alle bedeutenden Ergebnisse des Prozesses in einem Dokument zusammengefasst und können somit maßgeblich zur Optimierung im Entwicklungsprozess beitragen.
Mayhew liefert mit dem Usability-Engineering-Lifecycle eine sehr ausführliche Beschreibung eines Usability-Engineering-Prozesses, die sich sehr an den Erfordernissen der Praxis im Unternehmen orientiert. Neben der Darstellung der Phasen und Aktivitäten werden auch verschiedene Rollen beschrieben sowie die dazugehörigen Aufgaben und Verantwortlichkeiten. Der Usability-Engineering-Lifecycle wurde in der Arbeit untersucht, weil die Autorin bei ihrem Ansatz ebenfalls mit Rollenkonzepten arbeitet. So lassen sich Parallelen zum Prozess der Porsche AG herstellen.

3.1.2 Contextual-Design

Die Kontextanalyse ist das Kernstück des *Contextual-Design* Ansatzes. Diese verfolgt das Ziel, die Benutzer bei der täglichen Arbeit zu beobachten und zu interviewen. Beyer und Holtzblatt (1998) vertreten den Ansatz, dass die Struktur von Arbeit am besten am tatsächlichen Arbeitsplatz erfasst werden kann *(Context)*. Anschließend überlegt sich der Entwickler im Work-Redesign, wie die Benutzer sinnvoll unterstützt werden können. Hierzu werden mit strukturiertem Brainstorming Storyboards entwickelt. Im System Design werden die Details ausgearbeitet und überlegt, welche Funktionen der Benutzter braucht. Im Anschluss daran werden Prototypen entwickelt. Beyer und Holtzblatt (1998) gehen in ihrem Buch aber auch über die reine Beschreibung der Usability hinaus, indem sie Aspekte der Zusammenarbeit zwischen *Usability-Engineers* und anderen Abteilungen im Unternehmen wie beispielsweise dem Produktmanagement und den Entwicklern ansprechen und indem sie Anleitung geben, wie ihr Vorgehen an projektspezifische Umstände angepasst werden kann.
Beyer und Holtzblatt (1998) beschreiben in ihrem Buch, wie die unterschiedlichen Rollen, beispielsweise das Produktmanagement und die *Usability-Engineers*, zusammenarbeiten. Für den HMI-Entwicklungsprozess der Porsche AG sind die Voraussetzungen ähnlich (vgl. Kapitel 2). Auch hier beteiligen sich unterschiedliche Rollen im Entwicklungsprozess. Allerdings

beschreibt das *Contextuel-Design* nur die frühe Konzeptphase eingehend, der Entwurf der Benutzerschnittstelle wird nicht behandelt.

3.1.3 Usage-Centred-Design

Der Prozess des *Usage-Centered-Design* nach Constantine und Lockwood (1999) (vgl. Abbildung 3.1) knüpft an die traditionell objektorientierte Software-Entwicklung in Form von *Use Case* an. Mithilfe von sogenannten *essential use case*, die unabhängig von einer Software-Lösung sind, wird eigens für das *Usage-Centered-Design* eine Notation eingeführt.

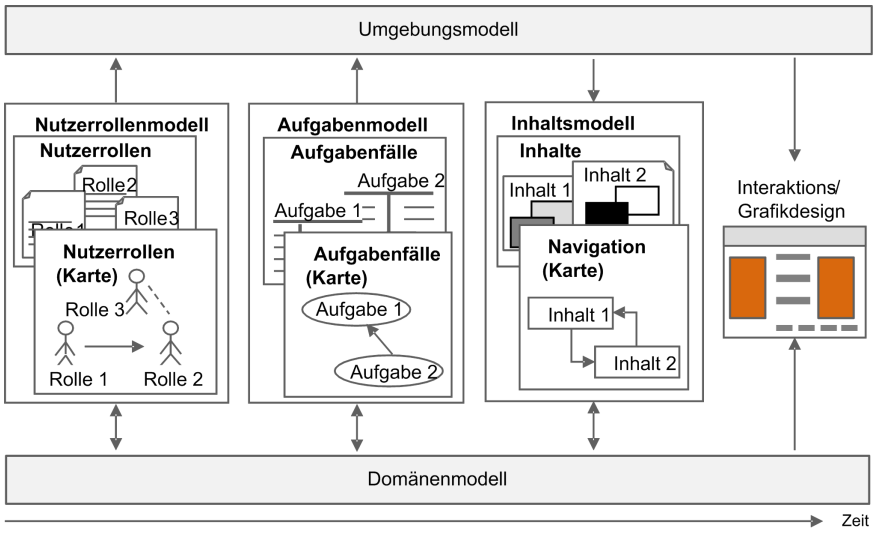

Abbildung 3.1: Usage-Centered-Design Constantine und Lockwood (1999)

Wie schon aus dem Namen *Usage-Centered-Design* Prozess hervorgeht, fokussiert sich der Prozess auf die Aufgaben der Benutzer. Die Problembeschreibungen und Lösungsvorschläge werden getrennt voneinander behandelt. Der modellbasierte Prozess lässt sich in User Roles, Use Cases und das Content Model aufteilen. Diese Modelle werden wieder mithilfe von *User Role Map*, *Use Case Map* und *Navigation Maps* beschrieben.

3.1.4 Useware-Entwicklungsprozess

Der Entwicklungsprozess von Zülke et al. (1999) wurde speziell für die Ansprüche der *Bediensystemgestaltung* in der Industrie entwickelt. Zülke et al. führte 1999 den Begriff Useware ein. Dies ist ein Sammelbegriff für alle Hard- und Software- Komponenten, die der Benutzung dienen (Zülke et al. 1999). Der Porsche HMI-Entwicklungsprozess, der in Kapitel 5 ausführlich beschrieben wird, baut auf dem Useware-Entwicklungsprozess auf. Dieser Prozess findet im Rahmen der Arbeit besondere Beachtung. Zülke beschreibt ein evolutionäres Vorgehen mit einer sehr frühen Nutzerintegration (Zülke et al. 1999). Der gesamte Prozess ist in *Analyse*, *Strukturgestaltung*, *Bediensystemgestaltung* und *Realisierung* unterteilt (vgl. Abbildung 3.2). Darüber hinaus verdeutlicht das Schaubild die entwicklungsbegleitende Evaluation während des gesamten Prozesses.

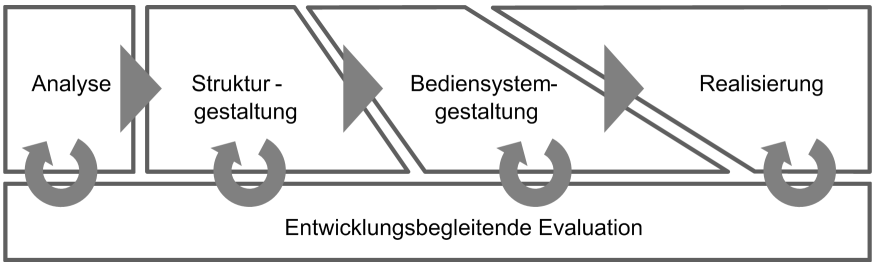

Abbildung 3.2: Useware-Entwicklungsprozess nach Zülke et al. (2004)

Zu Beginn, in der *Analysephase*, werden die Nutzeranforderungen erhoben, um die Ansprüche und Bedürfnisse der späteren Nutzer zu ermitteln. Ebenso erfolgt die Erfassung der technischen Anforderungen an die Bediensysteme (Bock 2007). Im nächsten Schritt der *Strukturgestaltung* wird ein sogenanntes Benutzungsmodell erzeugt, das zu diesem Zeitpunkt noch plattformunabhängig ist. In diesem Modell ist die grobe Interaktion des Nutzers mit einem Bediensystem abstrakt beschrieben. So können die mentalen Modelle der Nutzer bei der *Bediensystemgestaltung* berücksichtigt werden (Bock 2007). Im Anschluss an die *Strukturgestaltung* folgt die *Bediensystemgestaltungsphase*. Diese Phase ist in eine Grob- und eine Feingestaltungsphase unterteilt. Im Laufe der Phasen wird das Benutzungsmodell stetig optimiert. Dies erreicht man durch eine Konkretisierung der Anforderungen an das Bediensystem. Das Modell wird auf die entsprechende Plattform angepasst. Darüber hinaus erfolgt aus der Auswertung bisheriger Ergebnisse die Erstellung eines Bedienkonzeptes. Zuletzt leitet man in dieser Phase aus der Grobgestaltung die Feingestaltung ab, in der die Konzepte nach ergonomischen Kriterien konkretisiert werden (Bock 2007, Zülke 1999). Im vierten Schritt folgt die *Umsetzung* des zuvor spezifizierten Konzepts. Wie im Schaubild zu erkennen ist, erfolgt eine ständige Evaluation entwicklungsbegleitend. Dadurch werden spätere Nutzer in den Useware-Entwicklungsprozess integriert, um so anhand von Prototypen das entwickelte Konzept abzusichern (Zülke 2004).

3.1.5 Erkenntnisse aus den Entwicklungsprozessen

Die im Rahmen dieser Arbeit vorgestellten Entwicklungsprozesse haben nach genauer Untersuchung mehr oder weniger die gleichen Zielsetzungen. Hierzu zählt zum einen die frühe Einbindung der Nutzer, zum andern die Erstellung eines Nutzungskontextes auf Basis der Benutzeraufgaben (Burmester et al. 2005). Bei fast allen Entwicklungsprozessen lassen sich die Regeln von Gould und Lewis (1985) erkennen, die ein iteratives Vorgehen, die frühe Fokussierung auf Nutzer- und Aufgabenanforderungen sowie die empirische Überprüfung der Entwürfe durch Nutzer fordern.
Die Generierung von Prototypen erfolgt in einer sehr frühen Entwicklungsphase, um so eine entwicklungsbegleitende Evaluation zu gewährleisten. In den Entwicklungsprozessen werden unterschiedliche Methoden benutzt, um die Nutzeranforderungen abzuleiten. So benutzt der *Usability-Engineering-Lifecycle* von Mayhew (1999) in der ersten Phase der Anforderungsanalyse für die Erstellung von User Profiles Interviews und Fragebögen, um die späteren Nutzer und Aufgaben zu identifizieren. Der Lifecycle gibt jedoch dem Entwickler keine Unterstützung, wie er den Fragebogen bzw. die Interviews gestalten soll, um ein qualitativ

hochwertiges Ergebnis erzielen zu können. Die Vorgehensmodelle beschreiben lediglich die Methoden. Die Entwickler könnten hier beispielsweise durch abgelegte Erfahrungen von anderen Mitarbeitern profitieren, die ebenfalls User Profiles für andere Produkte erstellt haben. Der Ansatz von Mayhew (1999), einen Produkt-Styleguide während des Prozesses zu entwickeln, in dem die Ergebnisse aus den Phasen abgelegt werden, bietet einen guten Anknüpfungspunkt für diese Arbeit. Somit können die Entwicklungsstände transparent gemacht werden.
Nach Oed et al. (2001) reichen die bloßen theoretischen Kenntnisse aus den oben vorgestellten Entwicklungsprozessen in der Praxis nicht aus. Es fehlt der direkte Bezug zu den Rahmenbedingungen im konkreten Arbeits- und Projektkontext aus Sicht des Entwicklers (Oed et al. 2001). Bei der Untersuchung von Oed et al. (2001) kritisierten die befragten Personen "*mangelnde*" und "*zähe*" Kommunikation und ungenügende Absprachen trotz der eingeführten Entwicklungsprozesse. Darüber hinaus ist die Projektplanung und Koordination der Phasen sowie die Erfahrung aus vorangegangen Projekten wichtig für die Entwickler (Oed et al. 2001). Eine ergänzende Komponente zu den Prozessen könnte das projektübergreifende Wissen wieder verwendbar machen und die Kommunikation unter den Entwicklern stärker fördern. Dieses Untersuchungsergebnis deckt sich mit den Aussagen der Entwickler bei der Analyse des Porsche HMI-Entwicklungsprozesses (vgl. Kapitel 2).
Genau an diesem Punkt setzt die vorliegende Arbeit an: Sie will den Entwicklungsprozess verbessern, indem ein Konzept erarbeitet wird, das den Entwickler bei der *Bediensystemgestaltung* zu unterstützen versucht. Dies muss eine der zentralen Aufgaben sein, denn immer kürzere Entwicklungszeiten und beschränkte Ressourcen verursachen einen Qualitätsverlust der entwickelten Produkte. Aus diesem Grund muss die Effizienz und Effektivität als Erfolgskriterium beachtet werden.

3.2 Qualitätssicherung für Prozesse und Produkte

Es gibt grundsätzlich zwei unterschiedliche Wirkungsbereiche von Normen um die Qualität des späteren Endproduktes zu erhöhen. Zum einen gibt es produktorientierte Normen, bei denen es um die Erhöhung der Usability eines bestimmten Produktes geht. Zum anderen gibt es prozessorientierte Normen, die dabei helfen Entwicklungsprozesse im Unternehmen zu etablieren, um so eine gute Usability der Produkte sicherzustellen. Die höhere Qualität des Entwicklungsprozesses wirkt sich implizit automatisch auf die Produktqualität des zu entwerfenden Systems aus (Hurtienne und Prümper 2006). Darüber hinaus ist das Ziel einer Verbesserung der Prozessqualität die nachhaltige Sicherung der Gebrauchstauglichkeit von Produkten (DATech 2006). Die Qualitätssicherung beginnt nicht mit der Verifikation am Ende der Entwicklung wie beispielsweise dem Testen, sondern bereits mit der Validierung der Anforderungen während der Entwicklung (DATech 2006). Um zu überprüfen, in welchen Bereichen die Unterstützungskomponente eingreifen kann, werden zum einen die Normen von Entwicklungsprozessen und zum anderen die Normen für Produkte betrachtet.

3.2.1 Prozessqualität

In diesem Abschnitt werden Normen behandelt, die benutzerorientierte Gestaltung interaktiver Systeme gewährleisten. Wie lässt sich mithilfe von Normen ein Entwicklungsprozess gestalten, der die Anforderungen und Empfehlungen von Bediensystemen erfüllen kann? Die DIN EN ISO 13407 (benutzerzentrierte Gestaltungsprozesse für interaktive Bediensysteme) beschäftigt sich mit der Vorgehensweise, ein benutzerzentriertes Usability-Engineering zu erstellen. Einer der Grundsätze ist die aktive Beteiligung der späteren Nutzer sowie eine klare Festlegung der Benutzer- und Aufgabenanforderung. Eine geeignete Funktionsaufteilung zwischen den Entwicklern und Werkzeugen muss ebenfalls berücksichtigt werden. Darüber hinaus wird eine Iteration der Gestaltungslösungen empfohlen. Folgende Abbildung 3.3 zeigt den Gestaltungsprozess nach der DIN EN ISO 13407.

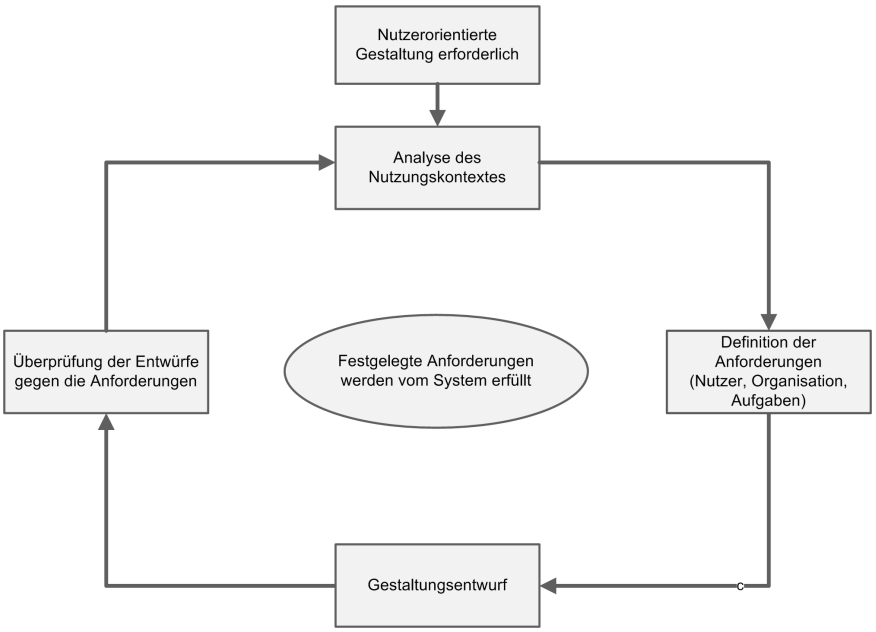

Abbildung 3.3: Schritte des DIN EN ISO 13407 Gestaltungsprozess für interaktive Systeme

Zusätzlich wird die frühe Einbindung von Entwicklern aus den verschiedensten Disziplinen (Rollen) befürwortet (Hurtienne und Prümper 2006, DATech 2006). Wie aus dem Software Engineering bekannten Capability Matury Model (CMM) Modell (Balzert 1998) haben sich in Anlehnung an die DIN EN ISO 13407 sogenannte Reifegrad-Modelle im Bereich des Usability-Managements entwickelt. Mithilfe der Modelle kann die Qualität des Usability-Engineering-Prozesses geprüft werden. Einer der bekanntesten Ansätze für die Bewertung von Prozessen ist das DATech-Prüfhandbuch für Usability-Engineering-Prozesse. Mithilfe des Prüfhandbuchs können Usability-Engineering-Prozesse anhand von definierten Beurteilungsdimensionen wie beispielsweise Dimension *14 Teamzusammensetzung* (Rollen) und *Benutzerdokumentation* in Reifegrade eingeteilt werden (DATech 2006). In den sogenannten Beurteilungsdimensionen sind bestimmte Tätigkeiten definiert. Diese müssen die erste Reifestufe abdecken, um überhaupt den Anspruch zu haben, in die nächst höhere Stufe zu kommen. In der ersten Stufe werden die definierten Aktivitäten zwar durchgeführt, diese sind jedoch nicht systematisch und ohne die Beteiligung von Experten des Unternehmens. In der zweite Stufe werden Usability Anforderungen validiert und zusätzlich werden Ex-

perten hinzugezogen. Darüber hinaus werden spätere Benutzter in den Prozess integriert. In der dritten Stufe ist das Usability-Engineering fest im Unternehmen verankert. Ferner haben sich Mechanismen zur ständigen Prozessoptimierung etabliert (DATech 2006). Das Vorgehen wird ausführlich am Beispiel des HMI-Entwicklungsprozess der Porsche AG in der Arbeit von Heilig (2007) beschrieben. Die Bewertung des Entwicklungsprozesses wurde ohne die Unterstützungskomponente, welche in dieser Arbeit konzipiert wird, durchgeführt.

3.2.2 Produktqualität

Immer häufiger kommen bei der Entwicklung von Bediensystemen ergonomische Normen und Standards zum Einsatz. Nach einer Studie von Reed et al. (1999) erhöht sich durch die Anwendung von Guidelines und Normen die Qualität der Bediensysteme deutlich. Aus diesem Grunde werden die Normen und Guidelines näher betrachtet, um diese in die Unterstützungskomponente zu integrieren. Zu Beginn wird jedoch der Begriff Konsistenz näher erläutert. Denn nach Normen (1988) ist die konsistente Gestaltung interaktiver Systeme das wichtigste Kriterium bei der Entwicklung, um so eine hohe Produktqualität zu erreichen.
Die konsistente Gestaltung von Produkten im Allgemeinen und Benutzerschnittstellen im Besonderen wird als eines der wichtigsten Kriterien benutzbarer Systeme erachtet. Vorteile liegen in der leichteren Erlernbarkeit, der Reduktion von Fehlern bei der Benutzung und der Verbesserung der Bedienbarkeit. Konsistenz betrifft nicht nur die grafische Darstellung, sondern ist in der Lage, die Gestaltung der Aufgabenstruktur, der Interaktionskonzepte und der Rückmeldungen an den Benutzer zu verbessern.
Abhilfe für die konsistente Gestaltung von Benutzeroberflächen verschafft die Verwendung von sogenannten Patterns. Mit der Entstehung der Objektorientierung fanden die Patterns zunehmend Anwendung im Bereich des Software Designs (Borchers 2001). Insbesondere im Bereich der Entwicklung von Mensch-Maschine-Schnittstellen fanden die HCI-Patterns schnellen Anklang. Derzeit gibt es einige sogenannte HCI-Pattern-Bibliotheken in den die Gestaltungsmöglichkeiten aufgezeigt werden.

Guidelines

Nach dem der Begriff Konsistenz geklärt wurde, wird in diesem Abschnitt erläutert, wie die Entwickler anhand von Guidelines bei der Gestaltung ergonomischer Benutzeroberflächen unterstütz werden können.
Vanderdonckt (1999) definiert Guidelines wie folgt: *"A guideline as any design and/or evaluation principle to be observed in order to get and/or guarantee the usability of a user interface (UI) for a given interactive task to be carried out by a given user population in a given context".* (Seite 82)
Nach Vanderdonckt (1999) können die Guidelines in *Design-Rules*, *Guideline-Sets*, *Standards*, *Styleguides* und *Ergonomic-Algoritmus* gegliedert werden. *Design-Rules* sind einfache User-Interface Spezifikationen, die keine weiteren Interpretationen beinhalten. Diese bestehen zum Beispiel aus Fenster-Templates. Die *Guideline-Sets* hingegen beinhalten Regel- und Art-Vorgaben für ein breites Feld der User-Interfaces. Solche Regeln werden beispielsweise in wissenschaftlichen Papers und Journalen veröffentlicht. *Standards* werden von nationalen

und internationalen Institutionen herausgegeben. Diese sorgen beispielsweise für eine Standardisierung der Bediensystemgestaltung im Fahrzeug, indem Symbole für die Anzeigen definiert wurden. Styleguides definieren das User-Interface-Design, indem sie Anhaltspunkte für die Entwickler bereitstellen. Dies geschieht nicht durch eine konkrete Regelbasis, sondern durch einen Art Empfehlung. Die meisten Firmen haben ihre eigenen Styleguides entwickelt. Die letzte Form von Guidelines ist die Art des *Ergonomic-Algoritmus*. Hierbei werden die Design-Aspekte formalisiert.
Guidelines und Normen verfolgen einerseits das Ziel des konsistenten *Look and Feels*, andererseits verbreiten sie auch das Usability-relevante Wissen. In der Praxis hingegen sind die Guidelines für die Entwickler schwer verständlich (Blume et al. 2005). Hier liegt die Schwäche darin, die Regel greifbar auf die Fragestellungen in den Projekten anzupassen. Der hohe Abstraktionsgrad der Regeln erschwert zusätzlich deren Anwendung. Meistens fehlen auch hier der Praxisbezug und der theoretische Hintergrund (Blume et al. 2005). Genau an dieser Stelle kann die Beratungskomponente ansetzten, indem die Regeln transparent dem Entwickler angeboten werden.
Limbourg und Vanderdonckt (2001) strukturieren Guidelines mit sogenannten virtuellen Karteikarten, um die Vielzahl der Guidelines in eine Struktur zu bringen. So ist beispielsweise der Zugriff auf die Kriterien mit einer Suchmaske möglich. Zu Beginn wird der Titel der Guideline eingetragen. Im Anschluss folgt eine kurze Beschreibung des Inhalts sowie die Referenz der Quelle (Buch, URL der Webseite). Die Organisation sowie der Hersteller der Guidelines werden ebenfalls in der Karteikarte abgelegt. Der Typ der Guideline und das Erscheinungsdatum sowie Beispiele der Anwendung sind weitere Punkte die Limbourg und Vanderdonckt (2001) aufnehmen, um die Guidelines in eine einheitliche Struktur zu bringen. Mithilfe dieser Struktur könnten beispielsweise die Guidelines sinnvoll für die Wissensdatenbank strukturiert werden, um so den schnellen Zugriff auf das entsprechend gewünschte Regelwerk zu gewährleisten.

Normen

Es gibt weltweit unzählige Institutionen, die regelmäßig Normen und Standards veröffentlichen. Zu den Bedeutendsten gehört das Deutsche Institut für Normung (DIN) und die International Organisation for Standardization (ISO). Speziell für Fahrerinformationssysteme gelten neben den zahlreichen Guidelines, die sich im Allgemeinen mit der Darstellung von Benutzeroberflächen am PC-Bildschirm beschäftigen, folgende Richtlinien: Alliance of Automobile Manufacturers (AAM)[1], ISO-Norm 15005 und 15008, Safe and Efficient In-Vehicle Information and Communication Systems (European Commission 1999), Association for Computing Machinery (ACM)[2].
Wie bereits erwähnt, gelten darüber hinaus bei der Bediensystementwicklung im Fahrzeug auch Normen und Standards der Gebrauchstauglichkeit für Mensch-Maschine-Schnittstellen wie beispielsweise die Teile 10 bis 17 der DIN EN ISO 9241. Der 11. Teil der DIN EN ISO 9241 beschreibt die Usability eines Produktes wie folgt: "Die Effektivität beschreibt die Erreichung der Nutzungsziele im vorgesehenen Nutzungskontext. Effizienz ist der eingesetzte

[1] http://www.autoalliance.org/safety/ (Lezter Zugriff: 12.08.2007)
[2] http://www.acm.org/ (Letzter Zugriff: 12.08.2007)

Aufwand, mit dem die Nutzer ein definiertes Ziel erreichen. Die Benutzerzufriedenheit gibt Aufschluss über die positive Einstellung der Benutzter zum Endprodukt." Für die Beratungskomponente zur Unterstützung der Entwickler müssen die einschlägigen Normen für Fahrerinformationssysteme betrachtet werden, denn anders als bei PC-Systemen kommen bei der Entwicklung dieser Systeme noch zusätzliche Normen und Standards zum Einsatz. So ist beispielsweise in der Norm ISO 15008 festgeschrieben, wie der Farbkontrast der Anzeigesysteme bei Tag- und Nachtfahrten sein sollte.

Grad der Automatisierung von Gestaltungskriterien

Um Gestaltungskriterien automatisch prüfen zu können, wie beispielsweise Styleguides und Normen, müssen diese in elektronischer Form vorliegen. Allerdings ist es sehr schwierig eine automatische Prüfung von Gestaltungskriterien durchzuführen, da diese sich sehr schwer formalisieren lassen. In Anlehnung an Scapin et al. (2001) und Beirekdar (2004) wird in Abbildung 3.4 der Grad der Automatisierung vorgestellt. Darüber hinaus werden anhand des PCM 2.1 die Bereiche aufgezeigt, in denen die Kriterien greifen.

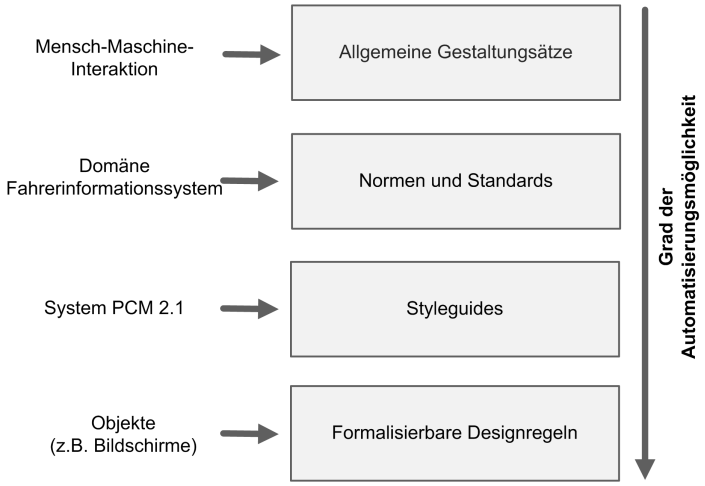

Abbildung 3.4: Grad der Automatisierung in Anlehnung an Beirekdar (2004)

Die *allgemeinen Gestaltungsansätze* bilden die Grundlage für die Normen und Standards. *Gestaltungsansätze* beinhalten die allgemeinen arbeitswissenschaftlichen Aspekte aus dem Bereich Mensch-Maschine-Interaktion. Diese sind jedoch oft sehr ungenau beschrieben, sodass eine relativ große Distanz zwischen den *Gestaltungsansätzen* und dem zu entwickelnden Produkt, wie beispielsweise dem PCM 2.1 liegt.
Normen und Standards sind meist auf eine Domäne (PCM 2.1) bezogen, im Gegensatz zu den *allgemeinen Gestaltungsansätzen*. Die Richtlinien sind jedoch sehr abstrakt formulliert, um so eine herstellerspezifische Unabhängigkeit zu erreichen. Dennoch sind die Richtlinien so konkretisiert, dass eine Abgrenzung zu anderen Domänen geschaffen werden kann zum Beispiel ISO 15008 *"Road vehicles- Ergonomic aspects of transport information and control systems - Specifications and compliance procedures for in-vehicle visual representation"*. Die Norm beinhaltet Richtlinien zur Gestaltung von Fahrerinformationssystemen. Die Ausprägungen der Richtlinie sind jedoch sehr stark abhängig von der Anwendungsdomäne, so gibt es beispielsweise eine Menge von Richtlinien zur Gestaltung von Bildschirmgeräten.

Neben den Kriterien der *Normen und Standards* spielen zum einen *Styleguides* und zum anderen die *Designregeln* eine sehr wichtige Rolle. Die Hersteller der Geräte stellen konkrete Regeln auf, um so eine Konsistenz in die Produktpalette zu bekommen. Um die *Corporate Design* zu gewährleisten, wird für jede Domäne ein *Styleguide* herangezogen, um so beispielsweise eine einheitliche Gestaltung der Oberfläche eines Fahrerinformationssystems zu erreichen. So dürfen beispielsweise beim Fahrerinformationssystem PCM 2.1 die Menüeinträge nur mittels eines orangenen Bandes gehighlightet werden.
Die *Designregeln* bilden die unterste Ebene. Diese werden von den Entwicklern systembezogen spezifiziert und lassen sich formalisieren. Dabei handelt es sich um detailliert ausformulierte Systemregeln, welche den Anwendungskontext und die Technologieaspekte beschreiben.

3.3 Benutzerunterstützung in Form von Hilfesystemen

Als Grundlage für die Arbeit werden in diesem Abschnitt Hilfesysteme von Softwareanwendungen betrachtet, um so die Charakteristika der Systeme herauszufiltern. Zusätzlich wird eine Klassifizierung von Hilfesystemen vorgenommen, um die verschiedenen Einsatzgebiete aufzuzeigen. Im Anschluss werden die verschiedenen Formen von Hilfesystemen erläutert.

3.3.1 Was sind Hilfesysteme?

Hilfe, die ein Hilfesystem anbietet, bezieht sich immer auf ein Anwendungssystem, von dem aus das Hilfesystem aufgerufen wird (Bauer und Schwab 1998). Bei der Entwicklung eines Hilfesystems muss entschieden werden, ob das Hilfesystem programmtechnisch unabhängig von einer Applikation ist, oder ob es in eine bestimmte Applikation integriert werden soll. Unabhängige Hilfesysteme können für verschiedene Applikationen verwendet werden und erlauben dadurch eine konsistente Handhabung für ein Hilfesystem, das zahlreiche Applikationen unterstützt (Balzert 1998).
Die wachsende Vernetzung von Computerarbeitsplätzen führt dazu, dass jedem Benutzer ein sehr umfangreiches Leistungs- und Funktionsangebot zur Verfügung steht (Bauer und Schwab 1998). Hat ein Benutzer ein Problem, so sollte das System ihm entsprechende Vorschläge unterbreiten, die die Anforderung des Benutzers befriedigen. Dabei sollte das System so intelligent sein, dass es lokal nicht verfügbare Leistungen über das Netzwerk von anderen Systemen abrufen kann. Die Autoren Bauer und Schwab (1998) unterscheiden Hilfesysteme in passive vs. aktive, statisch vs. dynamisch und einheitlich vs. individuell. Diese Systeme bilden die Basis für umfassende Unterstützungsleistungen für die Benutzer im Porsche HMI-Entwicklungsprozess. Deshalb werden sie im nächsten Abschnitt ausführlicher betrachtet.

3.3.2 Klassifikation von Hilfesystemen

In diesem Abschnitt erfolgt zunächst eine Klassifikation von Hilfesystemen. Darüber hinaus werden mögliche Formen von Hilfesystemen nach Shneiderman (1998) vorgestellt. Abbildung 3.5 gibt eine Klassifikation wider, die in Anlehnung an Bauer und Schwab (1998) erstellt wurde. Es lassen sich drei Ebenen für die Betrachtung von Hilfesystemen ableiten.

Passiv	**Aktiv**
Wird durch eine explizite Anfrage des Benutzers ausgelöst.	Wird ausgelöst, wenn das System feststellt, dass Hilfe benötigt wird.
Statisch	**Dynamisch**
Liefert Informationen, die den aktuellen Kontext nicht berücksichtigen.	Liefert Information, die den Kontext zum Zeitpunkt der Anforderung berücksichtigt.
Uniform	**Individuell**
Liefert für jeden Benutzer die gleiche Information.	Liefert Informationen, die an die speziellen Bedürfnisse der Benutzer angepasst sind (Rollenkonzept).

Abbildung 3.5: Klassifikation von Hilfesystemen

Initiative (aktiv vs. passiv)

Bei der Verwendung von Systemen kommt es zu Situationen, nach denen sich Benutzer im Rückblick wünschen, jemand hätte sie rechtzeitig auf eine fehlerhafte Verwendung des Systems aufmerksam gemacht. Zu lösen wäre dieses Problem durch ein Hilfesystem, das Ratschläge und Erklärungen geben könnte. Nach Suchman (1987) werden Computer nie imstande sein, jede auftretende Situation eines Menschen bei der Arbeit mit einem System zu erkennen. Selbst wenn es möglich wäre, alle möglichen Wörter und Aktionen aufzuzählen und dieses Wissen in einem System abzulegen, wäre dieser nicht in der Lage, die Informationen zu verstehen. Denn die zusätzlich benötigten Hintergrundinformationen sind nicht eingrenzbar (Suchman 1987). Diese Eigenschaft ist typisch für jede situierte Handlung von Menschen. Suchman (1987) verwendet hierfür den Begriff *situated action*. Damit meint sie, dass jede Aktion im Wesentlichen schnell und von den bearbeiteten Gegenständen und den sozialen Umständen abhängig ist. Darüber hinaus verweist Suchman (1987) darauf, dass ein menschlicher Partner in der Lage ist, das Umfeld der anderen Person mit einzubeziehen. Die wirkliche Arbeit der Benutzer ist für das System nicht sichtbar, wenn sie Symbole auf dem Bildschirm interpretieren. Das System erfasst lediglich die Resultate dieser Arbeit. Es hat daher einen stark eingeschränkten Blick auf die Arbeit der Benutzer. Diese Aussagen stimmen mit den Ergebnissen von mehreren Untersuchungen überein, die Aaronson und Carroll (1986) in ihren Arbeiten beschrieben haben. In diesen Untersuchungen wurden mehrere Versuche durchgeführt, bei denen Benutzer mit verschiedenen Systemen arbeiteten und Hilfe von unterschiedlichen Instanzen bekamen. Im ersten Versuch gaben Menschen persönliche Auskünfte. Im zweiten Versuch konnte Hilfe per E-Mail angefordert werden und im dritten bot ein Hilfesystem Unterstützung an, hinter dem sich ohne Wissen der Benutzer ein Mensch verbarg. Für das letzte Experiment wurden Hilfetexte vorbereitet, die die Person, die das Hilfesystem simulierte, an die Benutzer versenden sollte[3]. Darüber hinaus war es dieser Person erlaubt, in Ausnahmefällen individuelle Hilfetexte zu erzeugen. Wie sich herausstellte, mussten trotz sorgfältiger Erstellung der vorgefertigten Hilfetexte in über 55% der Fälle individuelle Hilfetexte verwendet werden, weil eine Reihe von Situationen eintraten, mit denen im Voraus nicht gerechnet worden war (Aaronson und Carroll 1986). Außerdem hatte die das Hilfesystem simulierende Person große Probleme, aus den Aktionen der Benutzer zu

[3]Diese Untersuchungstechnik, wenn ein Mensch ein System simuliert, wird Wizard-of-Oz genannt (Thevenin 1999).

schließen, ob und welche Hilfe erforderlich war.
Diese Probleme übertrafen bei Weitem die Schwierigkeiten der Experten, die den Benutzern in den anderen Versuchen zur Seite standen. So hatten die Benutzer selbst bei diesem *menschlichen* Hilfesystem ähnliche Probleme, wie sie auch bei Rechner gestützten Systemen verursacht werden. Die Hilfe kam zu spät, wenn die Lösung bereits gefunden worden war, die Benutzer warteten auf Hilfe und bekamen keine oder die Hilfe störte und wurde einfach sinnlos (Aaronson und Carroll 1986). Auch Budde und Züllighoven (1990) machen deutlich, dass ein Entwickler nicht vorhersehen kann, zu welchem Zweck ein Benutzer eine Applikation verwenden will. Daher sollten weder in einer Applikation noch in einem Hilfesystem Annahmen darüber getroffen werden, was der Benutzer mit seiner Tätigkeit beabsichtigt und ob der Benutzer damit noch ein sinnvolles Ziel verfolgt oder nicht (Robinson 1993). Aus den hier ausgeführten Argumenten lässt sich schließen, dass es keine Chance gibt, alle Situationen zu ermitteln, in denen Hilfe erforderlich sein kann, und dass die Unterstützungssysteme nur eine sehr eingeschränkte *Sicht* auf die Situation der Benutzer hat.

Kontextbezug (statisch vs. dynamisch)

Der große Nachteil statischer gegenüber dynamischer Hilfe besteht darin, dass das Programm dem Benutzer in jeder Anwendungssituation immer die gleiche Information anbietet (zum Beispiel Microsoft Office Hilfe). In diesem Fall verhält sich das Hilfesystem wie ein Handbuch mit erweiterten Suchfunktionen und verbesserten Darstellungsmöglichkeiten. Um dieses Potenzial von interaktiven Systemen auszunutzen, sollte die Handhabungskomponente eines Hilfesystems in der Lage sein, adaptiv auf das Wissen zurückzugreifen, das die von den Benutzern verwendete Applikation zur Verfügung stellen kann (Bauer 1988). Die dort erhältlichen Informationen beschränken sich normalerweise auf Angaben über den Zustand der Applikation und über die Aktionen der Benutzer bis zur Hilfeanforderung (Thevenin 1999). Eine Auswertung der Aktionen der Benutzer schließt sich aus denselben Gründen aus, wie das Erkennen einer Hilfesituation bei aktiven Hilfesystemen. Auch hier hat die eingeschränkte *Sicht* des Hilfesystems auf die Arbeit der Benutzer zur Folge, dass eine Prognose aller denkbaren Situationen unmöglich ist. Um dem Benutzer mehr als statische Hilfe anbieten zu können, ist es daher nur möglich, ein dynamisches Hilfesystem zu entwickeln, das Angaben über den Zustand der Applikation verwendet, mit der der Benutzer arbeitet (Bauer 1999).

Individualität (einheitlich vs. individuell)

Einheitliche Hilfesysteme bieten jedem Benutzer dieselbe Hilfe unabhängig von deren Kenntnisstand und Erfahrung. Individuelle Hilfe dagegen unterscheidet entweder verschiedene Gruppen von Benutzern oder fertigt die Eigenschaften eines jeden Benutzers individuell an (Rollenkonzept). Selbst wenn individuelle Hilfe ein wünschenswertes Konzept wäre, könnte die Handhabungskomponente eines Hilfesystems diese Forderung nicht ohne weiteres erfüllen (Duffy et al. 1992). Die Einteilung von Benutzern in Gruppen ist schwierig, weil die Wissensstände in verschiedenen Arbeitszusammenhängen zu große Unterschiede aufweisen. Die Alternative, alle Eigenschaften von Benutzern individuell einzuschätzen, ist auch schwierig, weil die Informationen für diese Einschätzung aus sehr begrenztem Material gewonnen werden müssen.

Eine weiterführende Art individueller Hilfe sind adaptive Systeme, die sich an die Erfahrung der Benutzer anpassen (Thevenin 1999, Duffy et al. 1992). Diese Systeme ändern je nach Einschätzung des Benutzers die Menge der zur Verfügung gestellten Informationen. Diese Art von Hilfesystemen ist für die Benutzer sehr verwirrend, weil es passieren kann, dass ihnen in zwei gleichen Situationen unterschiedliche Erklärungen angeboten werden. Darüber hinaus sind diese Systeme nicht in der Lage, den Wissensstand von Benutzern individuell einzuschätzen. Systeme, die in diesem Sinn individuell vorgehen, sind daher wie aktive Hilfesysteme weder sinnvoll noch zu verwirklichen.
Wenn man die Betrachtungsweise umkehrt und den Benutzer entscheiden lässt, welche Information er braucht, kann man Hilfe auch als individuelle Hilfe betrachten. In diesem Fall entscheidet nicht das System, was einem Anfänger oder ein Experte zugänglich ist und was nicht. Der Benutzer entscheidet, welche Information er in Anspruch nehmen möchte (Duffy et al. 1992). So kann sich der Benutzer im Hilfesystem individuell mit Informationen versorgen. Weiterführende Möglichkeiten, Hilfe individuell zu gestalten, bestehen darin, dass der Benutzer auf Wunsch Notizen schreiben und Lesezeichen für wichtige Informationen anbringen kann. Auf diese Weise kann der Benutzer die angebotenen Informationen selbstständig erweitern und sich eine eigene Sicht auf die Informationen erarbeiten. Anhand der hier angeführten Argumente lässt sich zusammenfassen, dass ein Hilfesystem individuelle Hilfe geben sollte. Dabei sollte der Benutzer nicht von der Systemseite aus eingeschätzt werden, sondern er sollte selbst entscheiden dürfen, welche Informationen er benötigt.

Auskunfts- und Beratungssysteme

Balzert (1998) klassifiziert Hilfesysteme in Auskunfts- und Beratungssysteme (vgl. Abbildung 3.6). Die Auskunftssysteme geben dem Benutzer Auskunft, über den Einsatz der Mensch-Computer-Schnittstelle und der Anwendungssysteme betreffenden Fragen.
Wobei hingegen Beratungssysteme den Benutzer über den Einsatz der Mensch-Computer-Schnittstelle und der Anwendungssysteme betreffenden Fragen beraten.

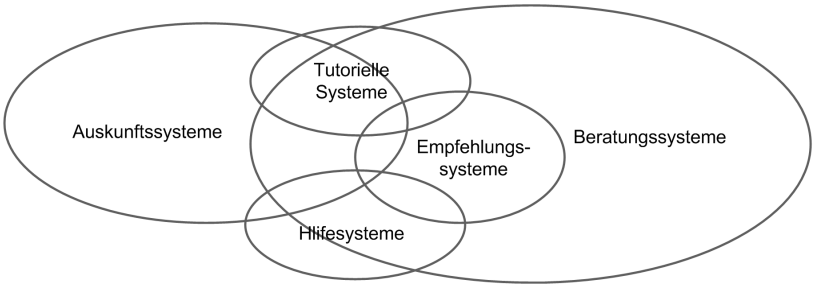

Abbildung 3.6: Auskunfts- und Beratungssysteme in Anlehnung an Balzert (1998)

Der Unterschied zu *klassischen* Anwendungssystemen besteht darin, dass sich keine direkten Arbeitsergebnisse erstellen lassen. Darüber hinaus unterscheiden sich die Unterstützungssysteme in qualitativer Art. Eine Beratungskomponente bietet dem Benutzer unter anderem umfassendere Unterstützung an als eine sogenannte Auskunftskomponente. Stellt man dem Benutzer eine Beratungsleistung zu Verfügung, so muss das System dem Benutzer wesentlich mehr *Know-how* dem Benutzer zur Verfügung stellen. Steht dieses Wissen nicht zur

Verfügung, kann keine qualitativ hochwertige Beratungsleistung erfolgen. Die beiden Komponenten Auskunft und Beratung überschneiden sich durchaus in einigen Bereichen. Ein Grund dafür kann die subjektive Einschätzung des Benutzers sein, ob er eine Unterstützungsleistung als Auskunft oder Beratung sieht. Balzert (1998) teilt die Auskunfts- und Beratungssysteme zusätzlich in sogenannte Teilsysteme auf. Zusätzlich definiert er Teilaufgaben (vgl. Abbildung 3.6). Dabei ist Balzert (1998) wichtig, dass sich die Systeme nicht gegenseitig ausschließen, sondern sogar Schnittmengen besitzen. Tutorielle Systeme helfen dem Benutzer beim Erlernen eines Anwendungssystem, beispielsweise einer Software. *Hilfesysteme* unterstützen bei der Benutzung eines Anwendungssystems. *Empfehlungssysteme* geben dem Benutzer bei der jeweiligen Anwendung eine Empfehlung ab. Das Konzept für die Unterstützungskomponente der Porsche AG sollte die Form der Beratung und Auskunft berücksichtigen, um so einen optimalen Mehrwert zu erzielen.

3.3.3 Formen von Hilfesystemen

Es gibt verschiedene Formen von Hilfesystemen. Diese beschreiben verschiedene Ansätze der Benutzerunterstützung. Shneiderman (1998) teilt die Systeme wie folgt ein:
Quick Reference fasst alle wichtigen Funktionen und Eigenschaften kurz und übersichtlich zusammen und lässt sich meist über Index und Suchfunktionen steuern. Dabei dient diese Hilfe eher als Erinnerungshilfe für bereits Erlerntes. Diese Hilfe ist meist kurz gehalten, um das schnelle Auffinden von relevanten Informationen zu gewährleisten. *Assistenten* zählen zu den aufgabenorientierten Hilfen. Durch den Assistenten wird der Benutzer schrittweise zur Lösung einer Aufgabe geführt. Dabei wird der Benutzer mit allen nötigen Informationen versorgt. Er hat jederzeit die Möglichkeit, Schritte rückgängig zu machen. *Online Dokumentationen* beschreiben die Anwendung im Layout eines ausführlichen Textes, der im Allgemeinen für lineares Lesen konzipiert ist. Oft werden hier die Druckausgaben der Handbücher in elektronischer Form bereitgestellt. Viele Großunternehmen mit hoch frequentierten Internetpräsenzen bieten Hilfesysteme mit fast zwischenmenschlicher Interaktion an. Unter *anthropomorphen Hilfesysteme* versteht man die Darstellung von Menschen oder Fabelwesen zur Überbringung von Hilfeinformationen. Dabei können die Charaktere durch Anklicken Videosequenzen abspielen, um den Benutzer durch ein Themengebiet zu führen oder bei entsprechenden Situationen über kleine Sprechblasen aktiv Hilfe anbieten. So wird eine Präsentation eines aktiven Hilfesystems dargestellt, indem das System die Aktionen des Benutzers beobachten muss und ihm bei Bedarf Hilfe anbietet. *Tutorials* sind in der Gruppe der dokumentenzentrierten Hilfesysteme zu finden und stellen eine Methode dar, dem Benutzer einer Anwendung dessen Funktionen und Aufgaben Schritt für Schritt zu vermitteln. Dabei stellt das Tutorial vorausgeplante Demonstrationen zur Verfügung, um den Benutzer durch bestimmte Szenarien zu führen. Die Form von Demonstrationen variiert zwischen Serien von Bildern, Video/Audio Demonstrationen oder die Abfolge von Texten. Der Benutzer spielt dabei meist eine passive Rolle. So verlangen Tutorials vom Benutzer, aufgrund der fehlenden Parallelität zur Anwendung, das Auswendiglernen von Szenarien, um diese später im Anwendungskontext ausprobieren oder direkt anwenden zu können. Die Lernphase befindet sich bei dieser Methode somit außerhalb der Anwendung. Die Problematik bei Tutorials liegt in der Aktualität der Inhalte gerade bei sich stets verändernden Applika-

tionen, wie Open Source Anwendungen oder WWW Services. *Hilfenetzwerke* sind in der Gruppe der computerzentrierten Hilfesysteme zu finden. Dabei setzt diese Form eine kommunikative Infrastruktur voraus. Durch diese Architektur hat der Benutzer die Möglichkeit, Fragen an einen oder mehrere Experten zu richten. Experten oder die Empfänger der Fragen können diese lesen und beantworten. So kann der Hilfesuchende binnen weniger Tage, aber auch Minuten oder gar nach Sekunden Lösungen zu seinem Problem erhalten. Dabei muss der Benutzer allerdings seine Unwissenheit innerhalb seiner Gruppe oder der Öffentlichkeit preisgeben. *Animationen* sollen den Neulingen einer Software die Grundfunktionalitäten und die Lösung von Aufgaben in Echtzeit näher bringen. So verfolgen Animationen die gleiche Zielsetzung wie Tutorials. Im Gegensatz zu diesen benutzen solche Hilfesysteme jedoch Videoausgabegeräte anstelle von textbasierten Medien. Aus konzeptioneller Sicht sind *FAQ's* ungeordnete Listen von am häufigsten gestellten Fragen bzw. Problemen, deren Stärke darin liegt, dem Benutzer in kurzer Zeit relevante Antworten zu seinen Fragen zur Verfügung zu stellen, ohne auf komplexere Hilfesysteme zurückgreifen zu müssen. So wächst durch FAQ's ein genaues Abbild der Systeme und Benutzerprobleme, die meist auf WWW Seiten zu finden sind. Dennoch entstehen in der Regel unorganisierte Listen, die es gerade dem Benutzer durch die ansteigende Anzahl von Einträgen das schnelle Finden von Lösungen frustrierend schwer machen.

3.4 Wissensmanagement

In den letzten Jahren wurde vielen Unternehmen bewusst, dass ihr Erfolg entscheidend von der Lernfähigkeit der Organisation abhängt (Huritenne und Prümmer 1998). Die Ursachen hierfür sind zum einen die verschärften globalen Wettbewerbssituationen und zum anderen die Verbreitung von neuen Kommunikationstechnologien. Das sogenannte Informationsmanagement umfasst das Managen der Informationen sowohl innerhalb als auch außerhalb des Unternehmens. Informationsmanagement dient wiederum dem Wissensmanagement auf allen Ebenen eines Unternehmens (Capurro 1998). Dieser Abschnitt bietet einen Überblick über die grundlegenden Theorien und Konzepte im Bereich des Wissensmanagements. Mithilfe des Wissensmanagements soll eine Infrastruktur geschaffen werden, welche die benutzerzentrierten Aktivitäten unterstützt. Das Know-how innerhalb des Unternehmens soll systematisch erfasst werden, um so den Entwicklern umfangreiche Beratungsleistungen zur Verfügung zu stellen.

3.4.1 Was ist Wissen?

Der Begriff des Wissens wird schon seit geraumer Zeit in den unterschiedlichsten Forschungsgebieten behandelt. Bis heute gibt es keine allgemeingültige Definition dieses Begriffes. Nach Puppe (2003) ist der Wissensbegriff: *"... ein Potential des Menschen, das ihn zum Treffen richtiger Urteile und zum zweckmäßigen Handeln befähigt".* (Seite 153)
Davenport und Prusak (1998) definieren Wissen wie folgt: *"Knowledge is a fluid mix of framed experience, values, contextual information, and expert insight that provides a framework for evaluating and incorporating new experiences and in formation. It originates and is applied in the mind of knowers. In organizations, it often be comes embedded not only*

in documents and repositories but also in organizational routines, processes, practices, and norms." (Seite 1)
Kaiser (1998) charakterisieren Wissen pragmatisch durch die unmittelbare Anwendbarkeit und Handlungsrelevanz. Auch Nonaka und Takeuchi (1998) gehen davon aus, dass der Unterschied zwischen Wissen und Information lediglich in der Unmittelbarkeit der Anwendung liegt.
Bei all diesen Definitionen wird der Wissensbegriff im Kontext von Handlung und Entscheidung erläutert. Wissen kann nicht direkt weitergegeben, sondern nur teilweise in Information transformiert werden. Der Empfänger interpretiert die Information und erzeugt daraus Wissen.
Der Faktor Wissen befindet sich nicht im Besitz des Unternehmens, er ist in den Köpfen der Mitarbeiter. Somit ist es für die Unternehmen eine große Herausforderung, ihre Mitarbeiter zur sogenannten Wissensarbeit zu motivieren. Die höchste Hürde dabei ist es, die Mitarbeiter zu sensibilisieren und eventuelle Vorbehalte gegen Wissensnutzung und -preisgabe abzubauen. Takeuchi und Nonaka (1998) unterscheiden in ihrem Modell zwischen explizitem und implizitem Wissen. Demnach ist explizites Wissen kodifizierbar und kann in formale Sprache übertragen werden. Implizites Wissen ist demgegenüber intuitiv, kontextspezifisch und daher kaum übertragbar und kodifizierbar. Die Schaffung von neuem Wissen liegt in der Umwandlung von implizitem in explizites Wissen mittels Analogien, Metaphern und Modellen.
Schwaninger (2000) gibt folgende Spezifikation an: "*Explizites Wissen umfasst Inhalte, die aufgrund bereits getroffener Unterscheidungen (Distinktionen) und durch Auswahlvorgänge (Selektionen) zustande gekommen sind. Implizites Wissen besteht in der Fähigkeit, Distinktionen und Selektionen, weitgehend intuitiv, laufend zu erkennen sowie diese in praktische Handlungen umzusetzen.*" (Seite 129)

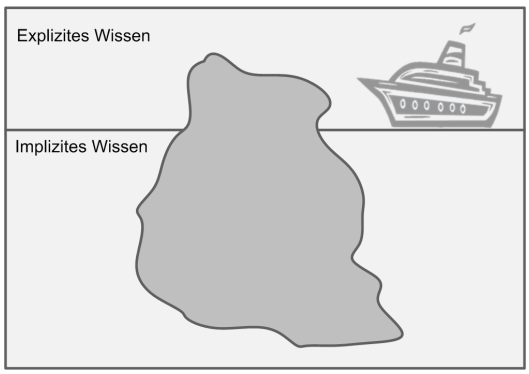

Abbildung 3.7: Explizites Wissen vs. Implizites Wissen

Analog dem Bild vom Eisberg im Wasser (vgl. Abbildung 3.7) wird davon ausgegangen, dass explizites Wissen nur einen kleinen Teil des in einer Organisation vorhandenen Wissens darstellt. Der größere, unsichtbare Teil wird im Bereich des impliziten Wissens vermutet. Der wichtigste Wissensträger mit dem größten Risiko ist der Mensch selbst: Als Einziger ist er in der Lage, aus Daten und Informationen Wissen zu gewinnen. Dieses kann er zu neuem Wissen kombinieren und auf unterschiedliche Anwendungssituationen übertragen. Gleichzeitig kann er als Einziger einen selektiven Umgang mit Wissen pflegen. Er kann

gezielt entscheiden, wann und wo er welches Wissen zur Verfügung stellt bzw. zurückhält (Schwaninger 2000). Abbildung 3.8 zeigt Beispiele für Wissensträger, unterteilt in implizites und explizites Wissen und je nach Anwendungsbereich dem internen oder externen Bereich zugeordnet.

Wissen	Internes Wissen	Externes Wissen
Implizites Wissen	Software Experten und Mitarbeiter	Zulieferer Beratungsfirmen
Explizites Wissen	Datenbanken Dokumentationen	Internet Publikationen

Abbildung 3.8: Beispiele für Wissensträger

Bohn (1994) teilt das implizite und explizite Wissen in ein Stufenmodell ein (vgl. Abbildung 3.9). Das Wissen wird von Stufe eins bis fünf leichter formalisierbar. Die Stufen drei, vier und fünf lassen sich problemlos in einer Wissensdatenbank ablegen, da sie formalisierbar sind. Für Stufe drei kann beispielsweise eine Art Arbeitsplatzbeschreibung in der Wissensdatenbank abgelegt werden. So können sich neue Mitarbeiter im Unternehmen schnell an die Gegebenheiten des Arbeitsplatzes anpassen. Das Unternehmen schont seine Personalressourcen, weil kein zusätzlicher Mitarbeiter zur Einarbeitung benötigt wird. Stufe vier könnte durch den HMI-Entwicklungsprozess der Porsche AG repräsentiert werden. Es könnten Workflows definiert werden, um so den genauen Ablauf zu koordinieren, wann beispielsweise Mitarbeiter A an Mitarbeiter B übergibt. In der Stufe fünf Know Why könnte der Stand der Technik formalisiert werden. Hierbei wäre das Ablegen von Guidelines und Normen denkbar.

Stufe des Prozesswissens	Beispiel für eine Ausprägung des Wissens	Beispiel für eine Präsentationsform des Wissens
1. Tacit Knowledge	Practical Thinking	Fremdbeobachtung
2. Embedded Knowledge	Kenntnisse von Routinen	Selbstaufschreibung
3. Know That	Kenntnisse der grundsätzlichen Arbeitsschritte	Stellenbeschreibung Arbeitsanweisung
4. Know How	Detaillierte Kenntnisse der Prozesslogik	Workflow
5. Know Why	Stand der Technik	Simulationsmodell

Abbildung 3.9: Stufen Modell nach Bohn (1994)

Mit der Einteilung von Bohn (1994) in die Stufen des Prozesswissens wird aufgezeigt, an welcher Stelle das Wissens sinnvoll eingesetzt werden kann. Um die Erfahrungen der Mitarbeiter im Unternehmen sinnvoll einsetzten zu können, müssen diese zunächst erfasst werden. Darüber hinaus werden Erfahrungen abstrahiert und generalisiert, um so die Erfahrungen auf möglichst viele Probleme anwenden zu können. Die Aufbewahrung von Erfahrung kann über sogenannte Wissensmanagementsysteme erfolgen. Die im System abgelegten Erfahrungen können für neue Projekte abgerufen werden. Dem Mitarbeiter müssen verschiedenen

Mechanismen bereitgestellt werden, wie beispielsweise eine Suchfunktion auf der Wissensdatenbank. Ferner müssen die ständig neu gewonnenen Erfahrungen den Mitarbeitern frei zur Verfügung stehen. Dies sind zum einen völlig neue Erfahrungen, zum anderen aber auch die Erfahrung über die Nutzung dieser Erfahrung, also eine Art Feedback ((Studer et al. 2001)). Durch diesen ständigen Verbesserungsprozess entwickelt sich ein Erfahrungskreislauf, der sich ständig optimiert.

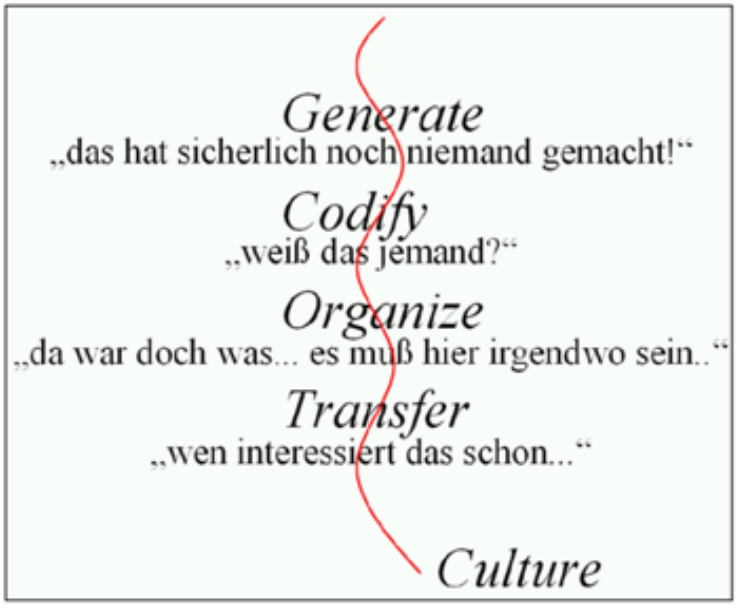

Abbildung 3.10: Praxisaspekte des Wissensmanagements (Studer et al. 2001)

Betrachtet man die Aufteilung von Wissensträgern nach der Delphi Group (1998), so ist das Wissensmanagement eine sehr große Herausforderung. Die größte Wissensbasis geht hierbei mit 42% von den Mitarbeitern des Unternehmens selbst aus. 26% des Wissens geht aus Papierdokumenten, so zum Beispiel Guidelines hervor. Die restlichen 32% liegen in elektronischer Form im Unternehmen vor. Der Studie nach kann bis zu 68% mehr Potenzial durch das Optimieren von Wissen generiert werden. Zusätzlich erreicht man durch die transparente Gestaltung von Wissen eine starke Optimierung im Entwicklungsprozess von Produkten, so die Delphi Group (1998).
Um das relevante Wissen entsprechend als Information zu archivieren und bei Bedarf zu Verfügung zu stellen, muss eine Architektur geschaffen werden, die den optimalen Zugriff auf das Wissensmanagementsystem gewährleistet. Die IT-Unterstützung ermöglicht eine beliebige orts- und zeitunabhängige Nutzung des Systems. Zusätzlich gestattet sie eine Unterstützung bei der Navigation und der Suche. Die Architektur des Werkzeuges muss sich an den Erfordernissen des Unternehmens orientieren. Es existieren derweil eine Reihe von Konzepten und Ansätzen die Wissenserzeugung im Unternehmen zu verankern. Eines der bekanntesten Modelle stammt von Probst (1998). Da dieses Model sehr starke Resonanz in der Literatur gefunden hat, wird es im Anschluss vorgestellt.

Das Modell nach Probst (1998) besteht im Wesentlichen aus acht Elementen. Er nennt es "*Bausteine des Wissensmanagements*" (vgl. Abbildung 3.11).

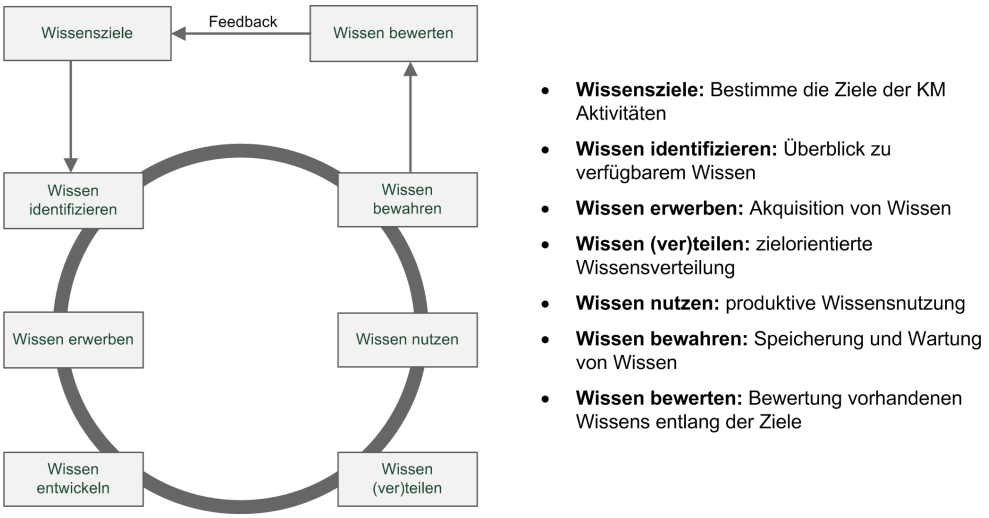

Abbildung 3.11: Prozessmodell nach Probst (1998)

Der Metaprozess des Wissensmanagements beginnt mit der *Wissensidentifikation*. Diese umfasst alle Maßnahmen, die sich auf die Schaffung von Transparenz über intern oder extern vorhandenes Wissen beziehen und stellt damit die Voraussetzung für den erfolgreichen Aufbau von Fähigkeitsdefiziten und die Schließung von Wissenslücken dar. Beim Wissenserwerb geht es um die Frage *"Wie erwerbe ich Wissen von außen?"*, also um den Erwerb von externen Experten. Möglichkeiten des *Wissenserwerbs* liegen im gezielten Abwerben oder Einkauf einzelner Personen, der Kooperation mit einem anderen Unternehmen oder dem Erwerb von Wissensprodukten.
Die *Wissensentwicklung* umfasst die Produktion neuer Fähigkeiten, besserer Ideen und leistungsfähigerer Prozesse. Wissensentwicklung beinhaltet alle Managementanstrengungen mit denen die Organisation sich um die Produktion von intern oder generell nicht existenter Fähigkeiten bemüht. Die *Wissensverteilung* behandelt die Fragestellungen *"Wer sollte was in welchem Umfang wissen oder können? Wie können die Prozesse der Wissensteilung erleichtert werden?"* Konkret meint dies das mechanische Verteilen von Wissenspaketen und den Austausch von Wissen auf der persönlichen Ebene.
Von zentraler Bedeutung ist der produktive Einsatz von Wissen, die *Wissensnutzung*. Hierzu gehört die nutzergerechte Darstellung von Wissen durch Visualisierung, Kurzzusammenfassungen aber auch durch eine nutzerfreundliche Arbeitsplatzgestaltung. *Erworbenes Wissen* dauerhaft nutzbar zu machen ist ein bedeutendes Ziel.
Wissensbewahrung meint daher den Schutz vor Wissensverlust. Damit gemeint ist nicht nur dokumentiertes Wissen, sondern auch der Erfahrungsschatz der Mitarbeiter. Um sich vor einer Informationsüberflutung zu schützen muss Wissen zudem aktuell gehalten werden und Veraltetes gelöscht werden. Die *Bausteine des Wissens* nach Probst (1998) stellen die möglichen Interventionsfelder für Wissensmanagementmaßnahmen im Unternehmen dar. Diese dienen als Konzept, um das Wissen im Unternehmen optimal zu nutzten. Greifen die Bausteine in der oben genannten Reihenfolge ineinander, so bildet sich ein Metaprozess (Probst 1998).

Eine Studie von Davenport und Prusak (1998) gibt Aufschluss darüber, weswegen Unternehmen aktives Wissensmanagement betreiben. Davenport und Prusak (1998) untersuchten 30 Unternehmen aus verschiedenen Branchen. Als Gründe für das Betreiben von Wissensmanagement nannten 35% der Mitglieder eines Unternehmens das Einbringen von Erfahrungen 29% nannten als Grund die Prozessverbesserung. Ferner gaben 23% der Befragten verschiedene Gründe an. Mit 13% gaben die Unternehmen das Skill-Management[4] an. Diese drei Kriterien mit einer Gewichtung von zusammen 77% zeigen das Verbesserungspotenzial im Entwicklungsprozess der Porsche AG.

3.4.2 Wissen als entscheidender Wettbewerbsfaktor

Wissen ist ein entscheidender Wettbewerbsfaktor und gilt damit als strategische Ressource für die Wettbewerbsfähigkeit. "Die Welt von heute ist nicht mehr arbeitsintensiv, sie ist nicht mehr werkstoffintensiv oder energieintensiv, sondern wissensintensiv" (Drucker 1992). Wissen trägt einen ausschlaggebenden Anteil zum Wertschöpfungsprozess heutiger Organisationen bei. Fest steht, dass man *Wissen nicht managen kann, aber man kann Umgebungen schaffen, in denen Wissen sich entwickelt und gedeiht* (Davenport und Prusak 1998). Es ist lediglich möglich, entsprechende Rahmenbedingungen zu schaffen, die Mitarbeiter dazu motivieren sollen, ihr Wissen zu teilen. Heutzutage sind die Entwickler der Porsche AG für eine Aufgabe im Entwicklungsprozess eines Bediensystems verantwortlich. Diese Gegebenheit hat wiederum eine hohe Abhängigkeit zur Folge: Das Wissen des Mitarbeiters bleibt solange im Unternehmen wie dieser dort beschäftigt ist. Was aber, wenn einer der Entwickler das Unternehmen verlässt? Sein Wissen verlässt dann mit ihm das Unternehmen. Dem kann Abhilfe geschaffen werden, indem man Lösungen findet, die das Wissen des einzelnen Mitarbeiters zum Wissen des Unternehmens machen. Auch in den aktuellen Prozessqualitätsbeurteilungen von DATech (2006) wird dieses Kriterium in der dreizehnten Dimension berücksichtigt. Ein Wissensmanagementsystem kann diese Dimension positiv beeinflussen, indem für jede Rolle im Entwicklungsprozess eine Arbeitsplatzbeschreibung als Modell abgelegt werden kann.

3.4.3 Barrieren beim Wissensmanagement im Entwicklungsprozess

Um Wissenstransfer systematisch strukturieren zu können, bieten sich bewährte Konzepte aus der Organisationstheorie an, wie beispielsweise Ablauforganisation, Koordination, Motivation und technische Infrastruktur. (Leitner und Tuppinger 2004). Nach Leitner und Tuppinger (2004) treten Barrieren für den Wissenstransfer beim zu transferierenden Wissen auf, beim Sender bzw. beim Empfänger des Wissens und im Kontext des Wissenstransfers. Abbildung 3.12 gibt einen Überblick über die Barrieren des Wissenstransfers. Ein Problem beim Wissenstransfer sind jene äußeren Faktoren, die verhindern, dass Beteiligte das erforderliche Wissen auch tatsächlich erhalten. Studien und Erfahrungsberichte über besondere Herausforderungen liefern ebenfalls wertvolle Erkenntnisse über zu berücksichtigende Einflussfaktoren und Maßnahmen (Probst 1998). Insbesondere mangelndes Vertrauen in das

[4]Skill-Managment befasst sich mit der Fragestellungen: "Wie wird das Wissen in den Köpfen der Mitarbeiter transparent für das Unternehmen"?

Wissen, fehlende Motivation bei Mitarbeitern oder kognitive Differenzen gelten als zu überwindende Barrieren des Wissenstransfers, aber auch mangelnde Funktionalität der verwendeten Software, die als Unterstützung des Wissenstransfers dienen sollte.

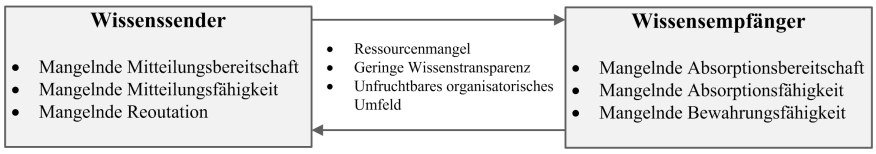

Abbildung 3.12: Barrieren beim Wissenstransfer (Leitner und Tuppinger 2004)

Der Grund, weshalb an dieser Stelle auf die Barrieren beim Wissenstransfer eingegangen wird, ist die Tatsache, dass dem gegenseitigen Austausch von Wissen elementare Bedeutung zukommt und daher Wege gefunden werden sollen, um dies bestmöglich zu gewährleisten und Störfaktoren zu minimieren (Fraunhofer Wissensmanagement Community 2004). Dafür ist es unbedingt nötig, mögliche Barrieren zu kennen, um ihnen entgegenwirken zu können. Nach einer Umfrage des Fraunhofer Institutes (Bügel 2004) geben die betreffenden Personen folgende Faktoren an:

- Keine Zeit
- Angst vor einer Blamage gegenüber Anderen
- Bedienerunfreundlichkeit der benutzten Software
- Geringes Verständnis für den erhofften Nutzen
- Angst vor Machtverlust (*Wissen ist Macht*)
- Angst, sich überflüssig zu machen (Arbeitsplatzverlust)
- Befürchtung der Reduzierung der Person auf Einträge in einer *Skill-Datenbank*

Die Barrieren des Wissensmanagements werden eingehend von Eberle (2003) erläutert. Diese Faktoren gilt es bei der Umsetzung im Unternehmen zu beachten.
Machtverlust: Mitarbeiter eines Unternehmens, die Wissen in Wissensdatenbanken einstellen, ermöglichen damit Kollegen den Zugriff darauf und dessen Verwendung. Insbesondere wenn Mitarbeiter in Konkurrenz um Aufstiegsmöglichkeiten stehen, werden sie Vorteile nicht gern durch Einstellung des Wissens in Datenbanken preisgeben.
Blossstellung: Wenn Mitarbeiter Wissen in eine Datenbank einstellen und damit veröffentlichen und zur Diskussion stellen, dann ist damit der Anspruch verbunden, dass das Wissen nicht trivial ist, sondern gewissen Wert und Seltenheit besitzt. Wenn Kollegen Wert und Seltenheit nicht so einschätzen, könnten sie ihre Geringschätzung auf die Person des Mitarbeiters übertragen.
Unsicherheit: Weniger erfahrene Mitarbeiter empfinden häufig Unsicherheit, ob ihr Wissen für Kollegen von Wert ist. Ihnen fällt es schwer einzuschätzen, ob Wissen zu allgemein und wohlbekannt ist, oder ob es sehr spezifisch für die spezielle Situation - und damit kaum übertragbar - ist. Die Einschätzung ist nicht trivial und erzeugt Unsicherheit.

Motivationsmangel: Wissen in eine Wissensdatenbank einzustellen bedeutet Mehrarbeit für die Zeit der Dokumentation und Beschreibung des Wissens. Die Nutzung von Wissensdatenbanken beruht unter anderem auf dem Prinzip, dass Mitarbeiter einerseits Mühe aufwenden und Wissen einstellen, um dieses anderen bereit zu stellen, andererseits davon profitieren, wenn sie selbst auf die Wissensdatenbank zugreifen. Wer diesem Prinzip des gegenseitigen Gebens und Nehmens nicht vertraut oder das Funktionieren noch nicht erfahren hat, dem fehlt ggf. ein Anreiz, die Mehrarbeit auf sich zu nehmen.
Sprachmangel: In manchen Unternehmen fehlt es an einer von allen verstandenen und akzeptierten Sprache, um persönliches Wissen und individuelle Erfahrungen auszudrücken und zu transportieren. Eine gemeinsame Sprache ist auch notwendig, um zum Beispiel mit gemeinsam verstandenen Metaphern und Analogien implizites Wissen auszudrücken und auszutauschen.
Hierarchie und Bürokratie: Streng hierarchische Unternehmensformen behindern Kommunikation und Wissensaustausch über die Grenzen von Unternehmensbereichen hinaus. Bürokratische Regelwerke und Formalia verhindern den Austausch von Wissen und Erfahrungen, dazu gehören zum Beispiel Vorschriften, wer Wissenswertes in einem Unternehmen auf welchem Weg verbreiten darf
Divergente Zielsysteme: Große Abweichungen und Gegensätze zwischen den Grundsätzen und Zielen eines Unternehmens und den individuellen Zielen der Mitarbeiter können dazu führen, dass Mitarbeiter individuelles Wissen, persönliche Erfahrungen und Einschätzungen nicht offen legen. Es besteht außerdem die Gefahr, dass individuelle Erfahrungen und Einschätzungen im Konflikt mit den Perspektiven des Unternehmens und den unternehmerischen Grundsätzen und Zielen stehen.

Kapitel 4

Werkzeuge zur Unterstützung von Usability-Entwicklungsprozessen

In diesem Kapitel werden Werkzeuge vorgestellt, die Entwickler bei der Gestaltung ergonomischer Bediensysteme unterstützen. Zu Beginn werden Werkzeuge zur Gestaltung von ergonomischen Benutzeroberflächen eingehend analysiert. Anschließend werden Werkzeuge zur automatischen Bewertung von Benutzeroberflächen vorgestellt. Den Abschluss der Analyse bilden Beratungswerkzeuge zur ergonomischen Gestaltung von Benutzeroberflächen.
Diese Untersuchung bildet die Basis für das Konzept von Unterstützungskomponenten für den HMI-Entwicklungsprozess der Porsche AG, welche im Rahmen dieser Arbeit konzipiert werden sollen. Es wird deutlich, dass die Modelle sehr unterschiedliche Ansätze haben, entsprechend den verschiedenen Zielen, die sie bei der Unterstützung des Gestaltungsprozesses der Benutzeroberflächen verfolgen. Allerdings wurden nur die im Rahmen dieser Arbeit bedeutsamen Werkzeuge zur Unterstützung herangezogen. Weitere Werkzeuge werden in der Literatur von Gomma et al. (2005), Ivory et al. (2001) und Reiterer (1995) vorgestellt.

4.1 Werkzeuge zur Gestaltung von ergonomischen Benutzeroberflächen

In diesem Abschnitt werden die Werkzeuge vorgestellt, die den Entwickler bei der Erstellung ergonomischer Benutzeroberflächen unterstützen. Nach Reiterer (2000) gibt es vier unterschiedliche Ansätze für die Gestaltung von Benutzeroberflächen: *Craft Approach*, *Cognitive Psychology* Approach, *Usability-Engineering Approach* und *Technology Approach*. Beim *Craft Approach* sind das Können und die Erfahrung der Entwickler gefragt, um ein qualitativ hochwertiges Ergebnis zu erreichen. Im Fokus stehen hierbei die Analyse, Implementierung und die Evaluation. Dieser Ansatz wird von keinem Werkzeug unterstützt. Somit sind hierbei Experten im Bereich *Usability-Engineering* notwendig. *Der Cognitive Psychology Approach* beruht auf der Theorie menschlicher Informationsverarbeitung. Der Fokus liegt auf der Analyse und der Evaluation. Beteiligte Personen sind nach Reiterer (2000) Psychologen und Usability-Engineer. Der *Usability-Engineering Approach* verbindet das Software Engineering mit dem User-Centered-Design. Wie auch schon bei dem vorherigen Ansatz steht die Analyse im Fokus. Der letzte Ansatz ist der *Technology Approach*. Die Philosophie

dieses Ansatzes ist die Automatisierung und Unterstützung des Software Engineering Prozesses. Dabei unterstützen die Werkzeuge, die diesen Ansatz verwenden, den Entwickler bei der Erstellung von grafischen Benutzeroberflächen (Reiterer 2000).

4.1.1 GENIUS

Mit *GENIUS* (**GEN**erator for **I**nterfaces **U**sing **S**oftware ergonomic rules) können Prototypen von Benutzungsoberflächen erzeugt werden. Das Projekt *GENIUS* wird vom BMFT[1] im Programm *Arbeit und Technik* unter dem Titel *"Unterstützungswerkzeuge zur benutzergerechten Gestaltung der Mensch-Computer-Schnittstelle"* gefördert. *GENIUS* versucht, den Entwurf ergonomischer Benutzungsoberflächen in existierende Methoden der Softwaretechnik zu integrieren (Janssen et al. 1993). Es handelt sich nicht nur um ein sogenanntes Vorgehensmodell, sondern um eine Werkzeugkette für die automatische Generierung von Benutzungsschnittstellen. Aus dem erstellten Datenmodell und der grafischen Repräsentation der Dialogabläufe werden erste Prototypen generiert.
Grundlage der Anwendung ist ein Datenmodell, dass in der Analysephase erstellt wird. Das Modell in Form eines Entity-Relationship-Diagramms enthält alle Daten, die der Benutzer sehen und bearbeiten möchte. In dem Beratungssystem *GENIUS* wird von einem sogenannten Sichtkonzept gesprochen, welches eine Erweiterung des Entity-Relationship-Diagramms ist. Ferner werden zu den Sichten textuelle Informationen in den Schemata festgelegt. Die Sichten berücksichtigen strukturelle Informationen, beschreibende Informationen und Informationen zu den Aufgabenmerkmalen. Aus diesen vordefinierten Informationen kann die Wissensbasis des Systems durch Anwendung von Regeln die Interaktionsobjekte der Benutzungsschnittstelle auswählen. Die Entwickler des Systems geben an, dass die Auswahlregeln nationale und internationale Standards berücksichtigen, wie beispielsweise die DIN 66234 1 bis 9. Eine Richtliniensammlung dient als Basis für die Aufstellung der Regeln. Als Grundlage dienen beispielsweise Guidelines von Smith und Mosier (1986), sowie Guidelines von verschiedenen namhaften Software-Herstellern. Das dynamische Verhalten der Benutzungsoberfläche wird durch sogenannte Dialognetze beschrieben. Die grafische Notation ist eine Anlehnung an Petri-Netze[2]. Die Dialognetze spezifizieren den Wechsel zwischen den Dialogen der modellierten Benutzungsoberfläche in Abhängigkeit von Funktionen aus der Sichtdefinition. Gemeinsam mit dem Layout der Benutzungsoberfläche entsteht so ein funktionstüchtiger Prototyp (Janssen et al. 1993). *GENIUS* soll eine Verbindung zwischen den üblichen, in der Software-Technik verwendeten grafischen Modellen zur Benutzungsschnittstellenentwicklung herstellen. Der Generierungsprozess ermöglicht eine schnelle Erstellung von Prototypen direkt aus der Spezifikation. Die konsistente Verwendung der Daten zwischen Anwendung und Benutzungsschnittstelle und die Einbeziehung von software-ergonomischem Gestaltungswissen in die Entwicklung ist ein großer Vorteil dieser Anwendung. Dabei soll die Verwendung expliziter Regeln für den Generierungsprozess eine hilfreiche Unterstützung

[1] Bundesministerium für Forschung und Technik.

[2] Ein Petri-Netz ist ein mathematisches Modell von nebenläufigen Systemen. Es stellt eine formale Methode der Modellierung von Systemen bzw. Transformationsprozessen dar. Ein Petri-Netz ist ein bipartiter und gerichteter Graph. Er besteht aus Stellen (Places) und Übergängen bzw. Transitionen (Transitions). Stellen und Transitionen sind durch gerichtete Kanten verbunden. Stellen werden als Kreise, Transitionen als Rechtecke dargestellt.

für die Einhaltung von Richtlinien und Style Guides und für die Einhaltung der Konsistenz bedeuten (Janssen et al. 1993).

4.1.2 JANUS

An der Universität Bochum wurde das System *JANUS* von Balzert (1994) entwickelt. Das System *JANUS* ähnelt dem System *GENIUS* sehr, denn beide Ansätze versuchen die Lücke zwischen Anforderungssystem und einem UIMS[3] (User-Interface Managementsystem) durch einen automatisierten Prozess zu schließen. Zusätzlich fließt bei diesen Systemen das sogenannte software-ergonomische Wissen mit ein. In verschiedenen Wissensbasen sind die Transformationsregeln gespeichert. Sie beinhalten Regeln für die Gestaltung von Layout, Dialogstrukturen, anwendungs- und benutzerspezifische Konventionen. Die Wissensbasis Layout enthält Regeln für den Fenster- und Maskenaufbau sowie deren Interaktionselemente. In der Dialog-Wissensbasis sind Regeln über Menüstrukturen und Menüarten verankert. Zusätzlich können benutzerspezifische Anwendungen festgelegt werden, wie zum Beispiel die Einhaltung von dialogspezifischen Gestaltungsregeln (Balzert 1994).
Laut Balzert (1994) haben erste Erfahrungen mit *JANUS* gezeigt, dass es möglich ist, Fachkonzepte als Ausgangspunkt für eine wissensbasierte Generierung von Dialogstrukturen zu verwenden. Der Vorteil dieses Systems soll darin liegen, dass der Benutzer verschiedene Dialogvarianten schnell durchspielen kann, indem er die Konvention modifiziert. Eine hohe Produktivität wird auch bei der Erstellung von Benutzeroberflächen erzielt.

4.1.3 MUSE II

Das Projekt *MUSE II* (**M**ethod for **U**ser **I**nterface **E**ngineering) wurde von Gorny an der Carl von Ossietzky Universität in Oldenburg entwickelt. Das Projekt sieht ein systematisches Design von Benutzeroberflächen vor. Zum einen soll nach Gorny (1997) die Entscheidungskomplexität und zum anderen die ergonomischen Prinzipien systematisch und dem Kontext entsprechend angewendet werden.
Das Vorgehensmodell von *MUSE II* sieht eine klare Trennung des Designprozesses in seinem Vorgehensmodell vor (vgl. Abbildung 4.1).

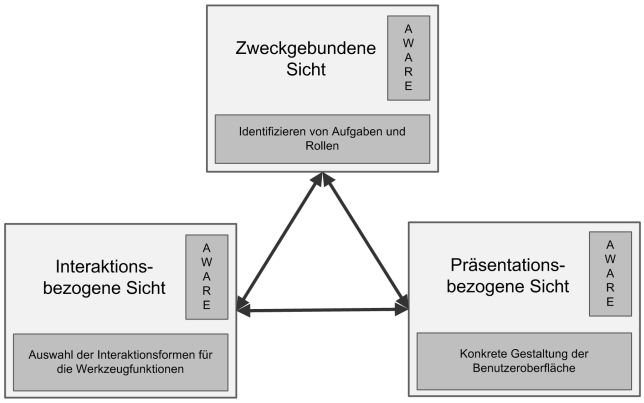

Abbildung 4.1: Softwareentwicklung mit MUSE (Gorny 1997)

[3]Unter der Bezeichnung User-Interface-Management-System (UIMS) sind Werkzeuge entwickelt worden, die im wesentlichen aus einem Compiler oder Interpreter für textuelle Beschreibungen von Dialogabläufen bestehen.

Der sogenannte Gestaltungsraum wird somit auf die gewünschte Sicht eingegrenzt, der Entwickler wird in der Entscheidungskomplexität unterstützt (Gorny 1997). Das aktuelle *MUSE II* Projekt basiert auf drei Sichten, wohingegen das MUSE I Projekt noch vier Sichten beinhaltete. *MUSE II* ist in *zweckgebundene* Sicht, *interaktionsbezogene* Sicht und *präsentationsbezogene* Sicht aufgeteilt. Der Designer kann jede der drei Sichten einnehmen, dabei werden fünf Designschritte empfohlen. Diese Phasen tragen den Namen *AWARE*:(1) **A**nforderung definieren, (2) **W**ahlmöglichkeit benennen, (3) **A**rgumente zusammenstellen, (4) **R**ichtlinien anwenden, (5) **E**ntscheidungen notieren.
Während des Designprozesses kann der Entwickler willkürlich zwischen den Sichten wechseln. Dies hat den Vorteil, dass der Entwickler freie Hand bei der Gestaltung der Oberfläche hat. In der *Zweckgebundenen* Sicht werden die Analyseergebnisse des Arbeitssystems umgesetzt. Zu dieser Sicht zählt beispielsweise das Identifizieren von Rollen. Als Ergebnis erhält der Benutzer eine Liste von Werkzeugfunktionen. In der zweiten Sicht der *Interaktionsbezogenen* Sicht, werden Entscheidungen über Interaktionsformen und Dialogstrukturen getroffen (Gorny 1997). Hier stehen dem Entwickler beispielsweise Grundformen von Ein- und Ausgabedialogen zur Verfügung. Ergebnis der Sicht sind Interaktionsformen und jeweils die Vorbedingungen für die Anwendbarkeit auf ein Objekt. In der letzen Sicht der *Präsentationsbezogenen* Sicht werden konkrete Erscheinungsformen für die Interaktionsform definiert (Gorny 1997).

4.1.4 EXPOSE

EXPOSE (**EX**pert System for **P**hase-**O**riented **S**oftware **E**rgonomic counseling) unterstützt den Entwickler bei der ergonomischen Entwicklung von Benutzungsoberflächen. Das Projekt wurde ebenfalls von Gorny an der Universität Oldenburg entwickelt. Das wissensbasierte Werkzeug hilft den Entwicklern während der Gestaltung von Benutzungsoberflächen, um so die ergonomischen Aspekte mit in den Prozess einfließen zu lassen. *EXPOSE* basiert auf der entwickelten *MUSE II* (vgl. Abschnitt 4.1.3), die ein systematisches Vorgehen erlaubt, um die Entscheidungskomplexität bei der Gestaltung zu reduzieren. Das System basiert auf vier unterschiedlichen Komponenten. In der ersten Komponente werden die bisherigen Erfahrungen, wie beispielsweise *work objects*, *related methods* und *tool functions* in einer Datenbank gesammelt (Gorny 1995). Das Expertensystem[4] im zweiten Schritt verfügt über softwareergonomische Richtlinien, die von bisher durchgeführten Untersuchungen im Bereich Human-Computer-Interaktion abgeleitet wurden. Die Richtlinien sind beispielsweise an *EVADIS*[5] (Oppermann et al. 1992) und *KABALeitfaden zur Kontrastiven Aufgabenanalyse*(Dunckel 1993) angelehnt worden. Im nächsten Schritt greift *EXPOSE* auf das Werkzeug *MUSE II* zurück, welches im vorherigen Abschnitt beschrieben wurden. Zusätzlich ist das System mit einer Hilfe ausgestattet, um so die entsprechend gewünschten Informationen zu *EXPOSE* zu bekommen (Gorny 1995).

[4] Als Expertensystem werden Software-Systemen bezeichnet, die auf der Basis von Expertenwissen zur Lösung oder Bewertung bestimmter Problemstellungen dienen.
[5] Leitfaden zur softwareergonomischen Evaluation von Dialogschnittstellen.

4.1.5 ProUSE

ProUSE (**Pro**cess Integrated **US**ability Engineering **E**nvironment) wurde von der Forschungsabteilung der DaimlerChrysler AG in Ulm in Zusammenarbeit mit der Universität Konstanz entwickelt. Das System soll den Prozess der Entwicklung von Benutzungsschnittstellen durchgängig unterstützen, deshalb ist es ein sehr interessanter Ansatz für das Konzept der Beratungskomponente in dieser Arbeit.
Dieser Ansatz wird *Evidence-based Usability-Engineering* genannt. Das System liefert eine Umgebung für die Analyse, die Spezifikation, den Entwurf, die Gestaltung sowie für die Bewertung bzw. Entwicklung von Benutzeroberflächen (Metzker 2005). So werden die Methoden des *Human-Centered-Design* verstärkt in den Softwareentwicklungsprozess eingebunden, um damit der Vielfalt von Anforderungen an die ergonomische Gestaltung von Benutzeroberflächen gerecht zu werden (Metzker und Reiterer 2002a).
ProUSE stellt eine Art Wissensdatenbank zur Verfügung. In ihr ist das nötige Wissen in sogenannten *USEPacks* abgelegt. Das Wissen ist semiformal beschrieben (Reiterer 2002). Während eines Projektes schlägt das System die jeweiligen Methoden vor, die auch die jeweiligen Rahmenbedienungen für genau dieses Projekt erfüllen (Metzker und Reiterer 2002b, Reiterer 2002). Die Selektion der Methoden geschieht über ein Filterwerkzeug. Darüber hinaus können auch die Benutzer Bewertungen für die Wissensinhalte abgeben. Diese werden anschließend vom System entsprechend berücksichtigt. Das auf den aktuellen Anwendungsfall passende Wissen kann beispielsweise in Form von ergonomischen Dialogbausteinen, Fragebögen oder Guidelines abgespeichert sein. ProUSE besteht aus mehreren Modulen (vgl. Abbildung 4.2). Diese sind das *ProUSE Portal*, *CAUSE* , *SATUP*, *CUES* und *RESUE*.

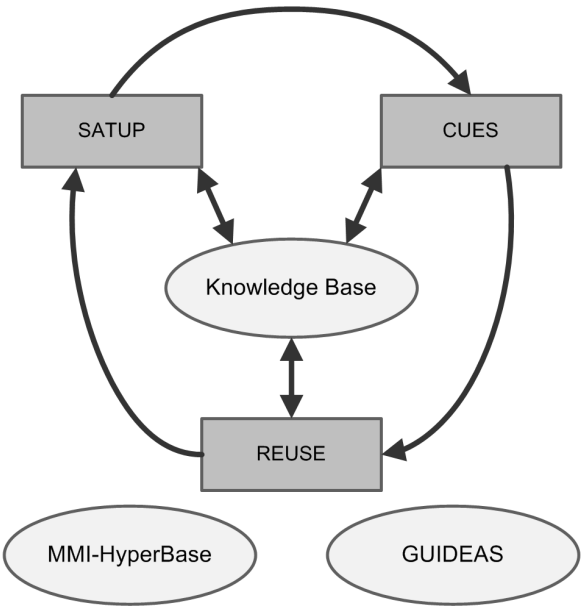

Abbildung 4.2: Komponenten von ProUSE (Metzker 2005)

Das *ProUSE Portal* ermöglicht den schnellen Zugriff auf die vorher genannten Komponenten. Durch die Webunterstützung des Systems ist der Zugriff standortunabhängig. Im Mittelpunkt der Module steht die Wissensbasis *CAUSE* (Evidence-based **C**omputer-**A**ided **U**sability-Engineering **E**nvironment) (Metzker 2005). Die Datenbank enthält die eingepfleg-

ten *USEPacks*, in denen die gesammelten Erfahrungen gespeichert werden und somit bei zukünftigen Projekten wieder verwendet werden können. *USEPacks* bestehen aus drei Teilen. Die Hauptinformationen wie Titel, Abstract, Beschreibungen usw. stehen in der *Core Information*. Zum einen beschreibt die *Context Situation*, die mithilfe des Kontextmodells generiert wird, den Entwicklungskontext des *USEPacks*. Zum anderen beschreiben die *Artifacts* beispielsweise Vorlagen für Benutzertests (Metzker 2005, Metzker und Reiterer 2002b).
Die Komponente *SATUP* (**S**etup **A**ssistan**t** for **U**sability Engineering **P**rocesses) unterstützt die benutzerzentrierten Aktivitäten im Rahmen des Projektmanagements, wie beispielsweise Budget, Projektdauer und die verfügbaren Ressourcen.
CUES übernimmt das aus *SATUP* angepasste Referenzmodell und erstellt daraus eine projektspezifische Arbeitsumgebung. So entsteht eine zentrale Datensammlung für jedes Projekt. Solche Daten können beispielsweise Dokumentenvorlagen und spezielle Anleitungen sein. Die Arbeitsumgebung sollte für den Entwickler keine unnötigen Informationen enthalten.
REUSE ermöglicht die Erweiterung und Pflege der Datenbank mit den im Projekt festgehaltenen Erfahrungen. Ferner hilft REUSE bei der Weiterentwicklung der aufgenommenen Erfahrungen.
In drei Schritten wird eine möglichst passende Methode für das aktuell zu entwickelnde Produkt definiert. Der Ablauf ist wie folgt: Zu Beginn wird eine passende *User-Centered-Design* Methode für aktuelle Projekte gewählt. Im zweiten Schritt erfolgt die Durchführung des Projektes, wobei hier die Erfahrungen, die während des Projektes gesammelt wurden, dokumentiert werden. Beim letzten Schritt werden die gesammelten Erfahrungen gespeichert. Diese können bei Bedarf für das nächste Projekt abgerufen werden.

4.1.6 ERGO-CONCEPTOR

Der modellbasierte Ansatz wurde von Moussa et al. (2000) an der Universität Tunis vorgestellt. Das Vorgehensmodell von *ERGO-CONCEPTOR* zur Erstellung von ergonomischen Benutzeroberflächen basiert auf einer umfassenden Requirementsanalyse sowie einer Prozess- und Aufgabenanalyse. Aus diesen Analysen wird in Form von Petrinetzen ein Modell abgeleitet. Dieses Modell muss manuell modelliert werden. Die Transitionen führen zu einem Zustandswechsel. Die Zustandsänderungen haben Auswirkung auf die Benutzeroberfläche. (Moussa et al. 2000, Hamacher 2006). Guidelines bilden die Basis des wissensbasierten Systems. Mithilfe dieser Komponente erfolgt eine Zuweisung von *Command Objects* zu Elementen für die Eingabe und *Informational Objects* zu Elementen der Ausgabe (Moussa et al. 2000). Im nächsten Schritt werden die Dialoge generiert. *ERGO-CONCEPTOR* ist nicht nur ein Werkzeug für die Oberflächengestaltung, sondern kann auch aufgrund der ergonomischen Regel der Wissensdatenbank als Bewertungswerkzeug verstanden werden.

4.1.7 Gegenüberstellung der Werkzeuge

Abbildung 4.3 zeigt, dass die Werkzeuge den Ansatz des Technologist Approach und des Usability Engineering Approach verwenden. Die Unterstützung der Werkzeuge ist jedoch sehr unterschiedlich. Dies kann beispielsweise in Form einer Projektverwaltung *(ProUSE)*

bis hin zurr Generierung von komplett funktionsfähigen Prototypen gehen.
Der Aufbau der Werkzeuge ist bei *JANUS* und *GENIUS* identisch. Die Modelle *GENIUS* und *JANUS* ähneln sich außerdem in ihrer Vorgehensweise sehr, denn sie arbeiten nach einem strikten Prozess. Beide bedienen sich eines Konstruktionsplanes (Gorny 1997). Darüber hinaus liegt das Hauptaugenmerk auf Modellen und oder generischen Schritten aus dem Software-Engineering, als auf den Benutzerschnittstellen, die eher dem Bereich der Software Ergonomie angehören. *MUSE II* und *EXPOSE* legen den Schwerpunkt dagegen auf die Unterstützung bei Designentscheidungen. Die Verwendung von Benutzeroberflächen ist letztlich ein kreativer Prozess. Aus diesem Grund sollten die Einschränkungen der Kreativität des Oberflächendesigners soweit wie möglich unterbleiben (Labinsky 1996). Die Ansätze werden teilweise mittels Kritik- und Beraterkomponenten unterstützt. Hingegen haben die modellbasierten Werkzeuge (*GENIUS* und *JANUS*) vorrangig die Generierung der Schnittstellensoftware bzw. wesentliche Teile davon aus abstrakten Beschreibungsmodellen zum Ziel. Darüber hinaus ermöglichen sie Entwicklern, Benutzungsschnittstellen auf einem sehr hohen Abstraktionsgrad zu beschreiben. Der Ansatz von ProUSE ist mit einem Wissensmanagementsystem vergleichbar. Das System unterstützt den Entwickler während des ganzen Entwicklungsprozesses in Form von maßgeschneiderten Wissensbausteinen zur Gestaltung von ergonomischen Benutzeroberflächen. Der *ERGO-CONCEPTOR* erstellt automatisch Softwareoberflächen unter zur Hilfenahme von Petrinetzen.

Beschreibung	GENIUS	JANUS	MUSE II	EXPOSE	ProUse	Ergo-Conceptor
Verfolgter Ansatz	Technologist Approach	Technologist Approach	Technologist Approach	Technologist Approach	Usability Engineering Approach	Technologist Approach
Unterstützung	Generierung von Schnittstellensoftware durch abstrakte Beschreibungsmodelle	Generierung von Schnittstellensoftware durch abstrakte Beschreibungsmodellen	Hilfe bei Design Entscheidungen von Software Dialogen	Hilfe bei Design Entscheidungen von Software Dialogen (baut auf MUSE II auf)	Prozesses Entwicklung von Benutzungsschnittstellen (Hypermedial)	Automatische Erstellung von Softwaredialogen mit Petrinetzen
Aufbau/ Modelle	Generierung statische sowie dynamische Spezifikation der Benutzeroberfläche mittels Entity-Relationsship-Modell) Wissensgenerator	Generierung der Spezifikation mit Anwendung von OOA Modellen , zusätzlich Janus Dialog- und Layoutexperte	Zweckgebunden Sicht Interaktionsbezogenen Sicht Präsentationsbezogenen Sicht	Kritik- und Beratungs-komponenten (Datenbank Expertensystem, MUSE II)	ProUSE , SATUP , CUES, REUSE	Manuelle Modellierung von Petrinetzen für interne Zustandsänderungen
Ausgabe	Prototypen von Benutzer-oberflächen für datenbankorientierte Anwendungen	Prototypen von kompletten Anwendungen (nicht nur Oberfläche)	Oberflächenentwurf (Fokus liegt auf Design)	Prototyp auf Basis von MUSE II	Projektmanagement Projektdokumentation Erfahrungen (USEPacks)	Bewertung und Ausgabe von Softwaredialogen

Abbildung 4.3: Werkzeuge zur Gestaltung von ergonomischen Benutzeroberfächen

Die Aufteilung von *MUSE II* in *zweckgebundene*, *interaktionsbezogene* und *präsentationsbezogene* Sicht ähnelt sehr der Aufteilung, welche von Bock (2007) für die Werkzeugkette aus dem Projekt *MORE* getätigt wurde. Bei den Werkzeugen *GENIUS* und *JANUS* steht eher die Erstellung von Prototypen im Vordergrund. *ProUSE* hingegen ist ein Ansatz, welcher das effiziente Zusammenarbeiten der verschiedenen Rollen (vgl. Abschnitt 2.3) am Entwicklungsprozess fördert. Darüber hinaus wird eine Ergonomie-Kompetenz im Unternehmen aufgebaut, von der, die am Entwicklungsprozess beteiligten Personen, profitieren können.

4.2 Werkzeuge zur automatischen Bewertung von Benutzeroberflächen

Grundlage einer Bewertung von Gebrauchstauglichkeit interaktiver Systeme auf Basis von Guidelines ist eine formale Beschreibung des Systems. Eine solche formale Beschreibung des Zielsystems ist beispielsweise in den *Porsche Ergonomic Rules* dargestellt (Porsche 2000). Weiterhin müssen die Kriterien in einer Form vorliegen, die eine elektronische Verarbeitung und Überprüfung der Benutzeroberflächen erlaubt.

4.2.1 ErgoVal

Mit Blickpunkt auf den strukturellen Aufbau einer Benutzungsoberfläche bietet das Werkzeug *ErgoVal* die Überprüfung auf Guideline-Konformität (Farenc 1997). Das Werkzeug bewertet die Benutzeroberfläche von Softwareprodukten. Nach Farenc (1997) konnte *ErgoVal* des Öfteren erfolgreich zur Bewertung und Verbesserung von Software-Dialogen bei der französischen Post eingesetzt werden. Zu Beginn folgt nach dem Einlesen des Softwaredialogs eine Art Dekomposition der Elemente, die sich auf dem Bildschirm befinden. Im zweiten Schritt werden die Bildschirmelemente nach einer fest vorgegebenen objektorientierten Struktur klassifiziert. Dabei sind mehrere Relationen der angegebenen Elemente zueinander möglich. So kann ein Element ein anderes enthalten. Diese örtliche Beziehung im Tool wird *Localisation relation* genannt. Die *Aggregation relation* gibt an, wenn ein Element eine oder mehrere Instanzen eines anderen Elementes enthalten kann. Die *Semantic relation* beinhaltet alle Elemente, durch die das Aussehen und das Verhalten eines Elementes beschrieben werden (Farenc 1997). Des Weiteren lassen sich den einzelnen Elementen der objektorientierten Struktur direkte Kriterien zuordnen. Diese können im Anschluss überprüft werden. Farenc (1997) hat dazu beispielhaft 205 Kriterien ausgewählt und in *ErgoVal* integriert. Ein Editor erlaubt die Erstellung und Bearbeitung dieser Kriterien.

4.2.2 KRI/AG

Löwgren und Nordqvist stellten 1992 das Werkzeug *KRI/AG* (**K**nowledge-based **R**eview of user **I**nterface) vor. Wird die GUI mit dem Werkzeug TeleUSE erstellt, so kann anschließend die Oberfläche einer Software bewertet werden. TeleUSE ist ein **U**ser-**I**nterface **M**anagement **S**ystem (UIMS), welches die volle Funktionalität von Motif-Programmen[6] zur Verfügung stellt (Löwgren und Nordqvist 1992). Dies geschieht über Werkzeuge wie WYSIWYG[8]-Editor, Oberflächen-Sprachinterpreter, Debugger usw. Die Bewertung kann aber nur bei X-Window-Systemen durchgeführt werden, dies schränkt die Anwendungsmöglichkeiten stark ein. In *KRI/AG* sind die Guidelines von Smith und Mosier (1986) und die Motif-Styleguides implementiert. Die in TeleUSE generierten Softwaredialoge werden in *KRI/AG* auf Konformität mit den Guidelines überprüft. Ein Bewertungsbericht teilt mögliche Fehler im *Oberflächen-* und *Informationsdesign* der Software mit (Löwgren und Nordqvist 1992). Die Basis der Bewertung von Oberflächen bildet ein sogenanntes Expertensystem. In diesem

[6]Motif ist eine grafische Programmbibliothek, mit der grafische Benutzerschnittstellen (GUIs) unter dem X-Windows-System[7] auf UNIX entwickeln werden können.

[8]Ist die Abkürzung für das Prinzip: "What You See Is What You Get"

sind die Regeln fest in Form einer proprietären Programmiersprache implementiert. Somit muss bei einer Eingabe oder bei eventuellen späteren Änderungen der schon bestehenden Regeln der Sourcecode direkt geändert werden.
Ein Vorteil des Werkzeugs von Löwgren und Nordqvist (1992) ist, dass auch sehr umfangreiche Dialoge bewertet werden können, weil die Implementierung der Guidelines als funktionale Regeln abgebildet werden.

4.2.3 Sherlock

Das Werkzeug *Sherlock* wurde vom Forth Institute of Computer Science in Griechenland im Jahr 2000 entwickelt. Es dient zur Verwaltung und automatischen Prüfung von Guidelines (Grammenos et al. 2000). Im Fokus von *Sherlock* steht die Bewertung des Oberflächendesigns. *Sherlock* bietet eine sehr umfangreiche Bewertung des Oberflächendesigns mithilfe von Guidelines an, deshalb wurde das Werkzeug in der Arbeit untersucht. Mit dem Werkzeug sind sehr umfangreiche Berechnungen von Oberflächenmaßen möglich. Einige der Metriken werden in Abschnitt 5.4.1 näher erläutert. Diese sind beispielsweise für die Berechnung der Oberflächenkonsistenz wichtig (Grammenos et al. 2000). Nach den Änderungen der Regeln muss der Source-Codes allerdings neu compiliert werden. Dementsprechend sind beim Benutzer umfangreiche Kenntnisse in einer der Programmiersprachen erforderlich. Dies ist von sehr großem Nachteil bei der Anwendung.

4.2.4 KWARESMI

KWARESMI wurde von Beirekdar (2004) an der Pariser Universität entwickelt. Das Werkzeug stellt dem Entwickler ergonomisch gesammeltes Wissen durch Guidelines in Form von Beschränkungen im HTML-Code zur Verfügung (Beirekdar 2004). Die in HTML beschriebenen Oberflächen können dynamisch jeweils von Guidelines geprüft werden, sofern diese in HTML-Elemente portiert werden können. Somit liegt das Hauptaugenmerk dieses Ansatzes auf der Bewertung von Websites. Folgende Abbildung 4.4 beschreibt die Architektur des *KWARESMI* Werkzeuges.
Die Guidelines werden in Form von *GDL-Statements* (Guideline Definition Language) strukturiert. Guidelines, Styleguide und Normen variieren sehr stark in ihren Formaten, deshalb müssen diese in ihrer Ausdrucksstärke formalisiert werden können. Mithilfe der *GDL-Statements* ist es möglich, Guidelines- und Normeninformationen in ausreichendem Umfang zu beschreiben. Beirekdar (2004) hat ein Framework entwickelt, welches folgende Aufgaben unterstützt: Das Framework zeigt den Benutzern, ob die Automatisierung einer Guideline *theoretisch* oder *praktisch* sowie *total* oder *partiell* möglich ist. Das Framework dient zusätzlich dazu, eine konsistente Strukturierung der Guidelines durchführen zu können (Beirekdar 2004). Das Framework durchläuft fünf Schritte, um eine Bewertung mithilfe der Guidelines durchführen zu können: (1) Ermittlung von interessanten Elementen, (2) Klassifizierung der ausgewählten Elemente zu Evaluations-Sets, (3) Definition atomarer Evaluations-Sets, (4) Definition einer Evaluations-Bedingung, (5) Evaluations-Set aufräumen. *KWARESMI* wandelt die Inhalte der Webseite in XML[9]-basierte Ausdrücke um. Im Anschluss werden die

[9]Die Extensible Markup Language (XML), ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien.

formalisierten GDL-Regeln auf die XML-basierten Informationen der Seite angewandt. Das Ergebnis der Evaluation ist ein Bericht über die Übereinstimmung zwischen der Regelbasis und der Webseite (Beirekdar 2004).

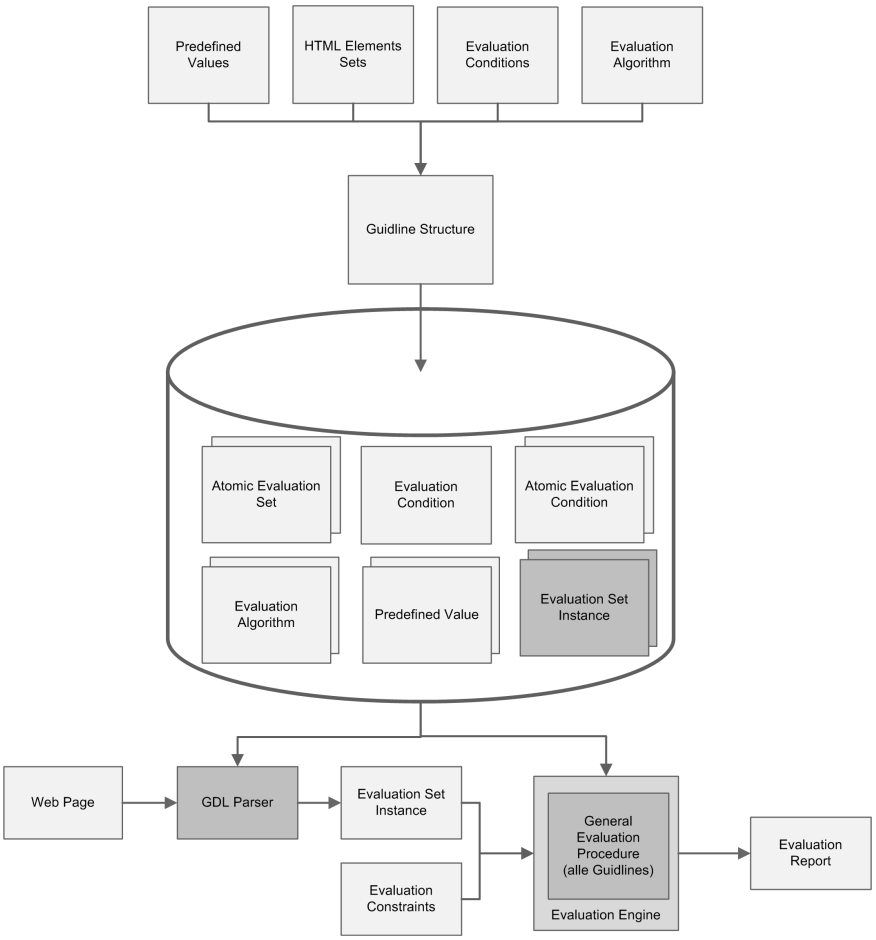

Abbildung 4.4: Konzept von KWARESMI (Beirekdar 2004)

4.2.5 ISO 9241-Evaluator

Die Evaluation mit dem *ISO 9241-Evaluator* dient zur Konformitätsprüfung mit der ISO-Norm 9241. Deshalb wird dieser Ansatz in der Arbeit genauer untersucht. Der *ISO 9241-Evaluator* ist die Weiterentwicklung des *EVADIS-* und des *EVADIS II-Verfahrens*, das zunächst zur ex-post Evaluation von fertigen Anwendungen gedacht war, jetzt aber zu einem Prüfverfahren gestaltet wurde.
Die Kriterien der Norm ISO 9241 werden durch ca. 500 Item-Sets dargestellt. Darüber hinaus enthalten die Sets detaillierte Begründungen des Kriteriums und dessen Überprüfung (Oppermann und Reiterer 1997). Die Benutzer können textuelle Beschreibungen und Screenshots zu den Sets eigenhändig hinzufügen. Als Ergebnis der Untersuchung erhalten sie einen Report mit zusätzlich angereicherten statistischen Daten (Oppermann und Reiterer 1997).

4.2.6 AIDE

1995 wurde zur Bewertung und Erstellung von Designvorlagen für Softwareprogramme das Werkzeug AIDE (**S**emi-**A**utomated-**I**nterface-**D**esigner und Evaluator) entwickelt (Melody et al. 2000). Die Maße (Fläche, die für die Informationsdarstellung benutzt wird, vertikale und horizontale Symmetrie und Verteilung der Elemente, Anzahl der Gruppen der Elemente) sind völlig operationalisiert (Sears 1995, Melody et al. 2000, Cherri et al. 2004). Die Maße werden gewichtet und zu einem Gesamtmaß addiert. Bedeutendster Faktor der Formeln ist die Berechnung eines besonderen Maßes. Dieses Maß berücksichtigt den Aufwand bei der Bedienung mit einer Maus auf dem Bildschirm. Sears (1995) nennt dieses Maß Layout Appropriateness. Mithilfe des Fitts´-Law wird errechnet, dass die Zeit, die benötigt wird, um mit dem Maus-Cursor ein Element auf dem Bildschirm zu erreichen, proportional zu dem Logarithmus aus dem Quotienten von der Größe und Entfernung des Elementes ist (Sears 1995). Für die Beschreibung der Dialogelemente benutzt *AIDE* das Textformat der Interface-Entwicklungsumgebung *Simple-User-Interface*[10] Toolkit.
Die Festlegung auf dieses wenig verbreitete Format schränkt die Anwendungsmöglichkeiten von *AIDE* deutlich ein. *AIDE* eignet sich daher nur für die Erstellung kleiner Eingabedialoge. Bedienungswahrscheinlichkeiten lassen sich nur für die Bearbeitung einer Aufgabe angeben (Sears 1995).
Der Hauptgrund, warum *AIDE* in der Arbeit betrachtet wurde, ist, dass es eines der wenigen Werkzeuge ist, das Formeln zur Berechnung der Oberfläche benutzt. Des Weiteren kann *AIDE* teilweise das *Interaktionsdesign* abprüfen, indem über das Maß *Appropriateness* die Angemessenheit des Layouts geprüft werden kann.

4.2.7 Gegenüberstellung der Werkzeuge

Die Werkzeuge haben alle eines gemeinsam: Sie dienen der Bewertung grafischer Benutzeroberflächen (vgl. Abbildung 4.5). Während die Werkzeuge *ErgoVal*, *KRI/AG*, *Sherlock*, *AIDE* und *KWARESMI* den Ansatz des *Technologist Approach* verfolgen, orientiert sich das Werkzeug *ISO 9241-Evaluator* am Ansatz des *Cognitive Psychology Approach*. Die dem *Technologist Approach* zugeordneten Ansätze, versuchen mithilfe von Werkzeugen den Engineering Prozess zu unterstützen. Dagegen verwendet der *ISO 9241 Evaluator* den *Cognitive Psychology Approach*. Dieser lehnt sich stark an die Theorie des menschlichen Informationsverarbeitungsprozesses an.
Die Eingaben der Regeln für die Bewertung erfolgen bei den Werkzeugen *ErgoVal*, *KWERESMI* und *ISO 9241-Evaluator* mithilfe eines Editors. Bei *KRI/AG*, *Sherlock* und *AIDE* müssen die Regeln dagegen in Form von Quelltexten angegeben werden. Somit ist eine Regelbearbeitung immer mit entsprechendem Programmieraufwand verbunden. Deutlich mehr Komfort für die nicht programmiererfahrenen Entwickler bieten die Werkzeuge *ErgoVal* und *KWAERSMI* und der *ISO 9241-Evaluator*. Die Anzahl der Guidelines und Normen ist sehr schwer abzuschätzen, da es mit mehr oder weniger großem Programmieraufwand möglich ist, sogenannte Item-Sets, wie sie viele der Werkzeuge benutzten, zu implementieren. Einige Autoren haben die Anzahl der verwendeten Regeln in ihren wissenschaftlichen

[10] Das Plattformunabhängige Werkzeuge Simple-User-Interface Toolkit wurde von der Universität Virginia entwickelt.

Veröffentlichungen angegeben. Die höchsten Item-Sets bietet mit ca. 500 das Werkzeug *ISO 9241-Evaluator*. Die Ausgabe der Ergebnisse findet meistens in textueller Art, in Form von Reports statt.

Beschreibung	ErgoVal	KRI/AG	Sherlock	KWARESMI	ISO 9241 Evaluator	AIDE
Verfolgter Ansatz	Technologist Approach	Technologist Approach	Technologist Approach	Technologist Approach	Cognitive Psychology Approach	Technologist Approach
Unterstützung	Bewertung der Benutzeroberfläche mit Focus auf strukturellen Aufbau	Bewertung der Benutzeroberfläche	Bewertung der Benutzeroberfläche mit Verwaltung der Guidelines	Bewertung der Benutzeroberfläche für Web-Seiten	Bewertung der Benutzeroberfläche nach Konformität ISO 9241	Bewertung und Erstellung von Design-vorschlägen
Eingabe der Regel von Normen und Guidelines	Editor Proprietäres Format	Kein Editor Programmier-Sprache C	Kein Editor Programmier-Sprache C Bibliotheken in Form von DLL	Editor Proprietäres GDL-Format	Editor	Kein Editor Programmier-Sprache C
Anzahl Guidelines und Normen	ca. 441	ca. 70	beliebige Anzahl	beliebige Anzahl	ca. 500 Item Sets	beliebige Anzahl
Ausgabe der Ergebnisse	Text mit Report	Text mit Report	Textuell mit Report	Textuelle Ausgabe Status :OK wenn Bedienung (Evaluation-Set) erfüllt sonst KO	Text mit zusätzlich statistischen Angaben	Ausgabe der Wahrscheinlichkeiten (textuell) wie ein Benutzer die Dialogelemente wechselt

Abbildung 4.5: Werkzeuge zur automatischen Bewertung von Benutzeroberflächen

Die größten Anknüpfungspunkte für das Konzept der Unterstützungsfunktion bietet das Werkzeug *KWARESMI* mit der implementierten *Regelsprache GDL*. Zusätzlich können die Benutzeroberflächen in XML beschrieben werden.

4.3 Beratungswerkzeuge zur ergonomischen Gestaltung von Benutzeroberflächen

Die am Entwicklungsprozess beteiligten Personen werden mithilfe von Guidelines unterstützt, um so die Gebrauchstauglichkeit der Systeme zu verbessern. Die Guidelines sind in den vergangenen Jahren immer umfangreicher geworden. In diesem Abschnitt werden Unterstützungswerkzeuge vorgestellt, die den Entwickler dahin gehend beraten, wie ergonomische Benutzeroberflächen gestaltet werden können.

4.3.1 IDA

Das Forschungsprojekt *IDA* (**I**nterface **D**esign **A**ssistance) beschäftigt sich mit der Fragestellung, wie Entwickler bei der ergonomischen Gestaltung von Benutzeroberflächen unterstützt werden können. Aus diesem Grund hat das Werkzeug eine besondere Relevanz und wird in der Arbeit tiefergehend behandelt.
Ziel des Projekts war es, gängige Entwicklungswerkzeuge zur Modellierung von grafischen Benutzeroberflächen zusätzlich mit Unterstützungsmöglichkeiten auszustatten (Reiterer 1995). Als Basis wurde das *User-Interface-Managementsyste*m der Firma ISA herangezogen. Dieses wurde mit der *IDA-Toolbar* erweitert. Das integrierte Werkzeug besteht zum einen aus einer objektorientierten Bibliothek von ergonomisch gestalteten Dialogbausteinen und zum

anderen aus einem hypermedialen Beratungssystem (vgl. Abbildung 4.3.1). Außerdem wird der Entwickler während des Entwicklungsprozesses mit einer wissensbasierten Qualitätssicherung unterstützt.

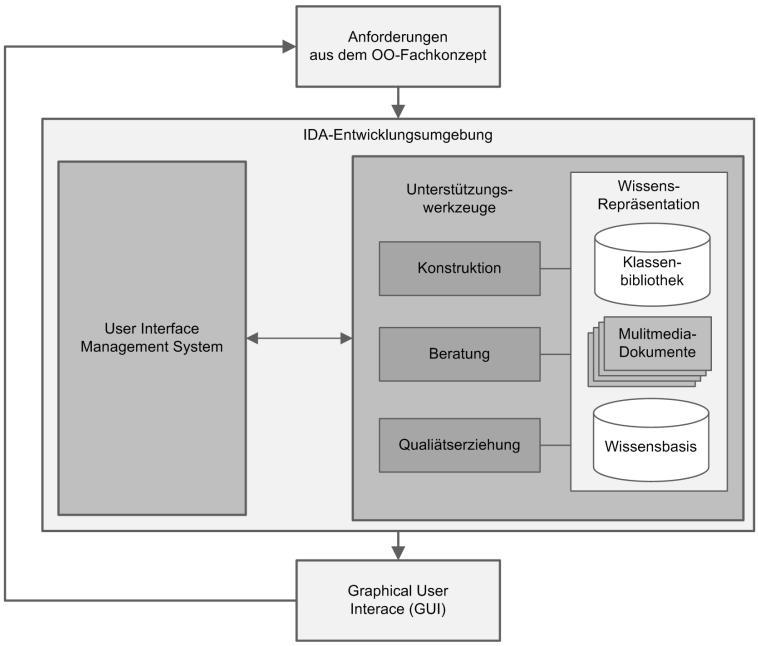

Abbildung 4.6: IDA Entwicklungsumgebung (Reiterer 1995)

Das Werkzeug unterstützt die Entwickler wie folgt (Reiterer 1995): Dem Entwickler werden während des Entwicklungsprozesses softwareergonomische Kenntnisse vermittelt *(learning on demand)*. Darüber hinaus können die Entwickler vorgefertigte ergonomische Dialogbausteine für eine Benutzeroberfläche verwenden *(reusability)*. Zusätzlich werden sie bei der Einhaltung von Standards und Richtlinien beraten (*usability*). Schon während der Implementierung können die Entwickler die ergonomische Qualität der Benutzeroberfläche automatisch prüfen lassen. Die ergonomisch gestalteten Dialogbausteine basieren auf Normen, Styleguides und herstellerunabhängigen Gestaltungsrichtlinien. Neben einfachen Dialogbausteinen, wie beispielsweise List-Box oder Tabellen, werden auch standardisierte Dialogabläufe, wie zum Beispiel Auswahl und Speicherungsdialoge, berücksichtigt. Das Beratungssystem unterstützt den Entwickler bei Fragen der ergonomischen Gestaltung bzw. bei Fragen zu grafischen Benutzeroberflächen. Mittels einer hypertextartigen Struktur wurden die Beratungsinhalte vernetzt. Die Inhalte können von einfachen Texten oder Grafiken bis hin zu Animationen oder Videosequenzen präsentiert werden. Mithilfe der wissensbasierten Qualitätssicherung können abgeschlossene Entwicklungsergebnisse auf grundlegende ergonomische und firmenspezifische Gestaltungsregeln automatisch geprüft werden *(quality assurance)*. Die gefundenen Fehler werden automatisch korrigiert und das Beratungssystem liefert die dazu gehörige Erklärung des Regelverstoßes (Reiterer 1995). Das Werkzeug bietet umfassende Möglichkeiten, um die ergonomische Qualität der Benutzeroberflächen zu steigern. Die Entwickler werden bei der Konstruktion von Dialogbausteinen durch eine Beratungskomponente, welche Ergonomiewissen bereitstellt, unterstützt. Außerdem beinhaltet das Werkzeug eine Komponente, die die Qualität der Benutzeroberfläche prüft. Das Werkzeug von Reiterer (1995) hat sehr viele Punkte, an denen die Unterstützungskomponente für die Porsche AG anknüpfen kann.

4.3.2 GUIDEAS

Das Unterstützungswerkzeug *GUIDEAS* (**GUI**dance for **DE**veloping **AS**sistance) soll den Entwickler von Assistenzsystemen während des Entwicklungsprozesses leiten bzw. unterstützen (Wandkte und Nitschke 2002). Das Werkzeug ist Teil des EMBASSI[11] Projektes. Dies hat das Ziel, Technik im privaten Bereich durch die Entwicklung von intelligenter Bedienassistenz für jedermann effizient nutzbar zu machen. Im Fokus des Expertensystems steht das *Informations-* und *Oberflächendesign*. Nach Wandkte und Nitschke (2002) soll *GUIDEAS* verhindern, dass der Entwickler nur rein technologiegeleitet - also sich nur an dem orientiert, was technisch machbar ist. Zunächst analysiert der Entwickler die Anforderungen an die Bedienassistenz, indem er Angaben über Benutzergruppen analysiert. Darüber hinaus werden Aufgaben, die Benutzer haben sowie die Nutzungssituation des Systems betrachtet. Diese Merkmale werden mithilfe von Abfrageformularen in Form von JAVA Applets erfasst. Zu allen Fragen werden Antwortalternativen vorgegeben, aus denen der Entwickler die zutreffenden auswählen kann. Verfügt er nicht über die erfragte Information, so kann er sich mithilfe einer empfohlenen Methode die fehlende Information beschaffen (vgl. Abbildung 4.7).

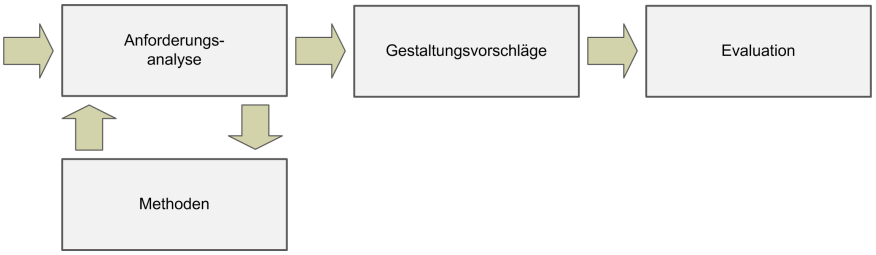

Abbildung 4.7: Vorgehensmodell zur Entwicklung von Assistenzsystemen Wandke et al. (2002)

Wenn die Anforderungsanalyse abgeschlossen ist, gibt das System einen Gestaltungsvorschlag ab. Dieser beschreibt, welche Funktionen Assistenz erfordern und welche Darstellungsmöglichkeiten im Hinblick auf *Oberflächen-* und *Interaktionsdesign* unterstützt werden sollen. Darüber hinaus geben die Autoren an: Hat der Entwickler seinen Vorschlag umgesetzt, so leitet *GUIDEAS* ihn durch die Phasen der Evaluation im Projekt. In der letzten Phase der Evaluation werden ausgehend von der Beschreibung des entwickelten Prototypen und der jeweiligen Evaluationsfragestellung geeignete Bewertungsmethoden vorgeschlagen (Wandke et al. 2002).
Der Ansatz von *GUIDEAS* richtet sich, wie oben beschrieben, an die Entwickler, die Anwendungen mit einer Assistenzfunktion ausstatten wollen. Hervorzuheben ist, dass die Entwickler in der Anforderungsanalyse sehr spezifische Angaben machen müssen. Kann der Entwickler aus irgendwelchen Gründen die Angaben nicht durchführen, wie beispielsweise aufgrund fehlender Informationen, so stellt das System ein Modul bereit, mit dem die fehlenden Daten erhoben werden können. Dieses System bietet im Bereich der Anforderungsanalyse gute Anknüpfungspunkte für die Unterstützungskomponente der Porsche AG.

[11]Das interdisziplinäre Forschungsprojekt EMBASSI ist eines der Leitprojekte zur Mensch-Technik-Interaktion des Bundesministeriums für Bildung und Forschung,in dem Ingenieure, Informatiker und Psychologen an der Entwicklung von intelligenter Bedienassistenz arbeiten.

Die Entwickler werden sehr gut bei der Bediensystementwicklung geführt.

4.3.3 Interface-Ratgeber

Der *Interface-Ratgeber* (Inra) wurde an der Humboldt-Universität Berlin entwickelt und ist ein hypertext-basiertes Informationssystem. Im System sind beispielsweise Gestaltungsregeln zu Dialogen enthalten. Die Informationen können lediglich manuell abgefragt werden. Der *Interface-Ratgeber* bietet dem Benutzer mittels des Hypertextsystems eine Verlinkung in den Guidelines an (Hüttner et al. 1995). So können beispielsweise Animationen und die dazugehörigen theoretischen Gestaltungsrichtlinien miteinander verlinkt werden. Nach Umfragen der Entwickler der Humboldt-Universität Berlin soll das Arbeiten mit dem System sehr motivierend sein, weil das Usability-Wissen erfolgreich angewendet werden kann (Hüttner et al. 1995). Der *Interface-Ratgeber* ist eher ein Werkzeug zur Guidelineverwaltung, da er keine Prüfung der Benutzeroberflächen berücksichtigt. Er wurde in dieser Arbeit dennoch betrachtet, da er die Informationen in einem hypertext-basierten Informationssystem darstellt, denn mithilfe der Hypertexte ist eine schnelle Navigation im System möglich.

4.3.4 GUIDE

GUIDE ("**G**uidelines for **U**sability through **I**nterface **D**evelopment **E**xperiences") wurde als eine Methode zum Einbinden von Guidelines in den Software-Entwicklungsprozess entwickelt. Die Anwendung von Usability Guidelines erfolgt kontextbezogen. Infolgedessen wird das Prinzip von Usability Patterns aufgegriffen (Henninger 2000). *GUIDE* verfolgt den Ansatz der Problemorientierung. Tritt ein Problem auf, wird dieses anhand von Attributen charakterisiert. Am Schluss werden unter der Anwendung eines sogenanntes Kontextmodells regelbasierte Vorschläge generiert. *GUIDE* besteht zum einem aus sogenannten Cases. Hier werden die bisherigen Erfahrungen gespeichert. Zum anderen aus sogenannten Rules, welche als Regelbasis dienen (Henninger 2000).

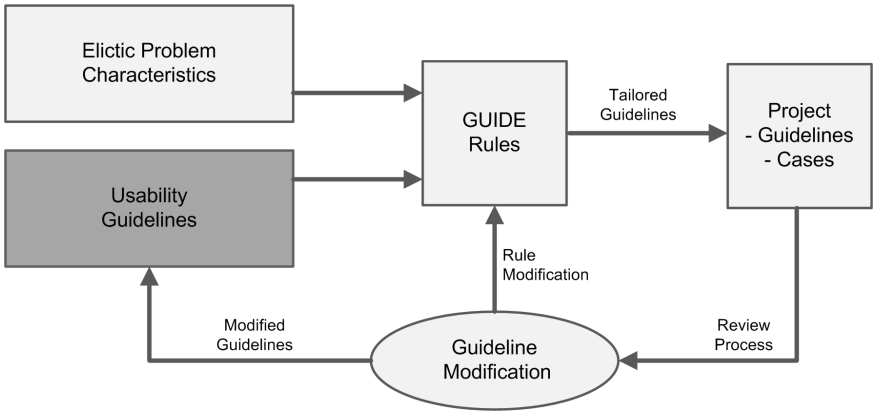

Abbildung 4.8: Der GUIDE Prozess (Henninger 2000)

Die *Cases* zeigen den Entwicklern, wie während eines vorherigen Entwicklungsprozesses bestimmte Guidelines konkret angewendet wurden. Die Entwickler haben somit Erfahrungswerte aus bisherigen Projekten. Darüber hinaus wird auch gezeigt, wie der Entwickler die Guideline erfüllt. *GUIDE* gibt eine Empfehlung für ein Anwendungsproblem ab. So greift

das System auf die Regel zu. Die Regeln sind als Fragen- und Antwortpaare in einem Entscheidungsbaum abgelegt. Je mehr Fragen der Entwickler beantworten kann, desto spezifischere Antworten werden vom System geliefert. Dieser Ansatz stammt aus dem Bereich der künstlichen Intelligenz und wird *Case-based reasoning* genannt. Das *Case-based reasoning* basiert auf der Annahme menschlicher Problemlösungsfähigkeiten (Henninger 2000). Henninger (2000) stellt zusätzlich noch eine Methode mit dem Namen *BORE* (**B**uilding an **O**rganizizanal **R**eposoitory of **E**xperience) vor. Auf dieses Projekt wird in dieser Arbeit nicht speziell eingegangen, da *GUIDE* die Basis bildet. Das Projekt *BORE* wird ausführlich in: "*Tool Support for Experience-Based Software Development*" von Henniger (2003) beschrieben. In diesem Ansatz könnten Anknüpfungspunkte für die Unterstützungskomponente im Bereich der Vorschläge, welches das System unterbreitet *(Case-based reasoning)*, liegen. So stellt das System aus den bisherigen Erfahrungen Lösungsvorschläge bereit, die Entwickler in die Bediensystemgestaltung einfließen lassen können.

4.3.5 Gegenüberstellung der Werkzeuge

Die größte Unterstützung für Entwickler bietet das Werkzeug *IDA* (vgl. Abbildung 4.9). Es vermittelt ergonomische Erkenntnisse in Form einer Wissensdatenbank, liefert vorgefertigte ergonomische Dialogbausteine und stellt mittels einer Komponente die ergonomische Qualitätssicherung fest. *Inra* und *GUIDE* dienen dabei vorwiegend der Verwaltung von Wissen bzw. Guidelines. *Inra* ist nur ein hypertext-basiertes Wissenssystem, in dem die Entwickler manuell beispielsweise nach Vorschriften bei der Entwicklung von Dialogsystemen suchen können.

Beschreibung	IDA	GUIDEAS	Inra	GUIDE
Verfolgter Ansatz	Technologist Approach	Cognitive Psychology Approach	Technologist Approach	Technologist Approach
Unterstützung	Ergonomische Erkenntnisse vermitteln, vorgefertigte ergonomische Dialogbausteine bereitstellen, ergonomische Qualitätssicherung	Assistent für Entwickler bei der Produkt-Gestaltung	Beratungswerkzeug für Entwickler durch gesammeltes ergonomisches Wissen	Beratungswerkzeug für Entwickler durch gesammeltes ergonomisches Wissen
Eingabe der Regel von Normen und Guidelines	Dialogbausteine C bzw. C++, Gestaltungswissen über Editor, Qualitätssicherung mit Hilfe von Dialog	Editor	Keine Editor Programmier-Sprache C Bibliotheken in Form von DLL	Rule Editor
Ausgabe der Ergebnisse	Gestaltungswissen textuell, Qualitätssicherung textuell mit Dialog	Textuelle Ausgabe zur ergonomischen Gestaltung	Textuelle Ausgabe	Textuelle Designvorschläge nach Regelbasis mit Dialog

Abbildung 4.9: Beratungswerkzeuge zur ergonomischen Gestaltung von Benutzeroberflächen

Zur lokalen Eingabe der Regeln verfügt *IDA* über einen Menüeintrag *Pflege*. Bei der Qualitätssicherungskomponente erfolgt die Pflege mittels eines Editors (Reiterer 1995). Dieser ist

im Expertensystem integriert. Mithilfe des sogenannten *Rule Editors* können die Entwickler in *GUIDE* neue Regeln anlegen. Die Werkzeuge greifen auf die textuelle Ausgabe von ergonomischem Wissen zurück. *IDA* stellt darüber hinaus die Qualitätssicherung textuell mit Dialogen dar. Einen ähnlichen Ansatz wählt *GUIDE*. Hier werden ebenfalls die Designvorschläge nach Regelbasis mit Dialog dargestellt. In der folgenden Abbildung 4.9 werden nochmals die Kriterien der Werkzeuge vorgestellt.

4.4 Erkenntnisse aus der Werkzeuganalyse

In diesem Abschnitt werden die Werkzeuge vorgestellt, die Anknüpfungspunkte zum Konzept der Unterstützungskomponente der Porsche AG aufweisen. Keines der Werkzeuge kann, nach eingehender Analyse, im aktuellen HMI-Entwicklungsprozess direkt eingesetzt werden, da sie den Anforderungen aus Kapitel 2 nicht gerecht wurden. Keines der Werkzuge schaffte über alle Prozessphasen den Entwickler so zu unterstützen, wie es für die Porsche AG sinnvoll wäre. Eine ausführliche Beschreibung der Unterstützungsmöglichkeiten wird im Anhang B in Form einer Matrix vorgestellt. Im Folgenden werden die Anknüpfungspunkte der Werkzeuge vorgestellt.
Der Ansatz von *IDA* (vgl. Abschnitt 4.3.1) eine Unterstützungskomponente in ein CASE-Tool zu integrieren, deckt sich sehr stark mit dem verfolgten Konzept, welches im Rahmen dieser Arbeit vorgestellt wird. Das Werkzeug bietet vielversprechende konzeptionelle Anknüpfungspunkte für diese Arbeit. Auch die Entwickler der Porsche AG sollen mit ergonomischen Bausteinen und Wissen sowie einer Art Qualitätssicherung versorgt werden.
Bei dem Ansatz von *KAWARESMI* (vgl. Abschnitt 4.2.4) werden Webseiten mithilfe eines Parsers in eine XML-basierte Form gebracht. Der Vorteil dieses Werkzeuges ist, dass nicht die gesamte Webseite geparst werden muss, sondern nur die Tags und Attribute, die anhand einer Guideline geprüft werden sollen. So können die Entwickler mit den zuvor definierten Evaluationssets (vgl. Kapitel 4) gezielt beispielsweise die Textfarbe eines Textlinks *(Set1 = Body. bgcolor, Body.link)* überprüfen (Beirekdar 2004). Dementsprechend soll auch für das Konzept der Porsche AG berücksichtigt werden, dass nur einzelne Kriterien geprüft werden können.
Der *Interface-Ratgeber* (vgl. Abschnitt 4.3.3) ist ein sehr einfaches Werkzeug und dient dem Entwickler als elektronisches Nachschlagewerk. Über die Hyperlinks können Themengebiete miteinander verbunden werden. Im Konzept des Wissensmanagementsystems müssen ähnlich wie beim *Interface-Ratgeber* die Informationen, wie Normen und Standards schnell gefunden werden können.
Eine etwas andere Form von Hilfe bietet das Werkzeug *ProUSE* (vgl. Abschnitt 4.1.5). Es unterstützt die Entwickler durch die Bereitstellung von prozessbegleitenden Methoden. So werden von *ProUSE* Wissensinhalte aus dem *Human-Centered-Design* empfohlen, die zuvor in Form von *UsePacks* definiert wurden. Die Entwickler können sogar die angebotenen Wissensbausteine bewerten. Ähnlich wie bei *ProUSE*, sollten die Entwickler Zugriff auf Erfahrungen von früheren Projekten haben.
Das Werkzeug *Sherlock* (vgl. Abschnitt 4.2.3) unterstützt den Entwickler sehr umfangreich im Bereich des Oberflächendesigns. Das Werkzeug gehört auch zu den wenigen, an

die die Konsistenzberechnung explizit adressiert ist, wobei auch bei *Sherlock* das direkte manipulative Eingreifen des Entwicklers nicht angeboten wird. Das bedeutet die Unterstützungsfunktion kann nur im Anschluss an die Designentwicklung herangezogen werden, um die Dialoge zu evaluieren. *Sherlock* bietet Methoden zur umfangreichen Evaluierung. Es wird aber an keiner Stelle erwähnt, wie diese Methoden in einen Entwicklungsprozess integriert werden können. Das Werkzeug *Sherlock* konzentriert sich eher auf die Rückmeldung der evaluierten Ergebnisse als auf eine integrierte Prozessunterstützung. Bei den Ergebnissen der Evaluierung werden häufig beim Report Durchschnittswerte berechnet, die jedoch nicht in die individuellen Elemente eingesetzt werden können (Mahjan und Shneidermann 1997). Im Bereich des *Informationsdesigns* bietet *Sherlock* sogenannte *Terminology Bags* an, in denen semantische und ähnliche Begriffe abgebildet werden können (Mahjan und Shneidermann 1997). Das System schlägt beim Auftreten von Synonymen automatisch die Verwendung eines einheitlichen Begriffes vor. Eine weitere Komponente der Anwendung, das sogenannte *Interface Concordance Tool*, erfasst alle Beschriftungen von Dialogen und zeigt dem Entwickler die Unterschiede in der Groß- und Kleinschreibung an. Die Entwickler werden bei konsistenten Beschriftungen von Dialogen unterstützt, indem die leichten Variationen aufgezeigt werden.
Die Überprüfung von semantischen Beziehungen, so zum Beispiel die Prüfung auf Synonyme, unterstützt keines der vorgestellten Werkzeuge, wobei die vorgestellten Werkzeuge ausreichende Anknüpfungspunkte für das Konzept der Beratungs- und Unterstützungskomponente bieten, welches im nächsten Kapitel näher erläutert wird.

Kapitel 5

Konzeption einer unterstützenden Komponente für den Porsche AG HMI-Entwicklungsprozess

In den vorangegangenen Kapiteln wurden die theoretischen Grundlagen sowie die aktuellen Werkzeuge zur Unterstützung der Entwickler aufgezeigt. In diesem Kapitel wird nun die Konzeption der unterstützenden Komponente *PoCKET* (**Po**rsche **C**onsulting and **K**nowledge **E**ngineering **T**ool) vorgestellt. Als Basis dient die Analyse aus Kapitel 2, in der der Porsche HMI-Entwicklungsprozess charakterisiert wurde.
Zunächst wird der aktuelle HMI-Entwicklungsprozess erläutert. Dabei stehen die einzelnen Prozessphasen im Mittelpunkt. Anschließend werden die Unterstützungsmöglichkeiten in den Prozessphasen aufgedeckt, um so detaillierte Anforderungen an die Unterstützungskomponente zu erlangen. Abschnitt 5.4 befasst sich mit dem Aufbau der Werkzeugunterstützung. Im Anschluss werden die Konzepte der Beratungskomponente, die in MetaEdit+ integriert wurden und das Konzept der Wissensplattform, welches in *Lotus Notes* integriert wurde, erläutert. Zuletzt wird der Mehrwert der Unterstützungskomponente anhand des DATech-Prüfhandbuches aufgezeigt.

5.1 Der HMI-Entwicklungsprozess der Porsche AG

Der HMI-Entwicklungsprozess der Porsche AG ist sehr stark an den Useware-Prozess von Zülke et al. (1999) angelehnt (vgl. Abschnitt 3.1.4). Abbildung 5.1 zeigt die Integration des Useware Entwicklungsprozesses von Zülke et al. (1999) in den Fahrzeugentwicklungsprozess der Porsche AG. Der Fahrzeugentwicklungsprozess lässt sich in *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung* unterteilen.

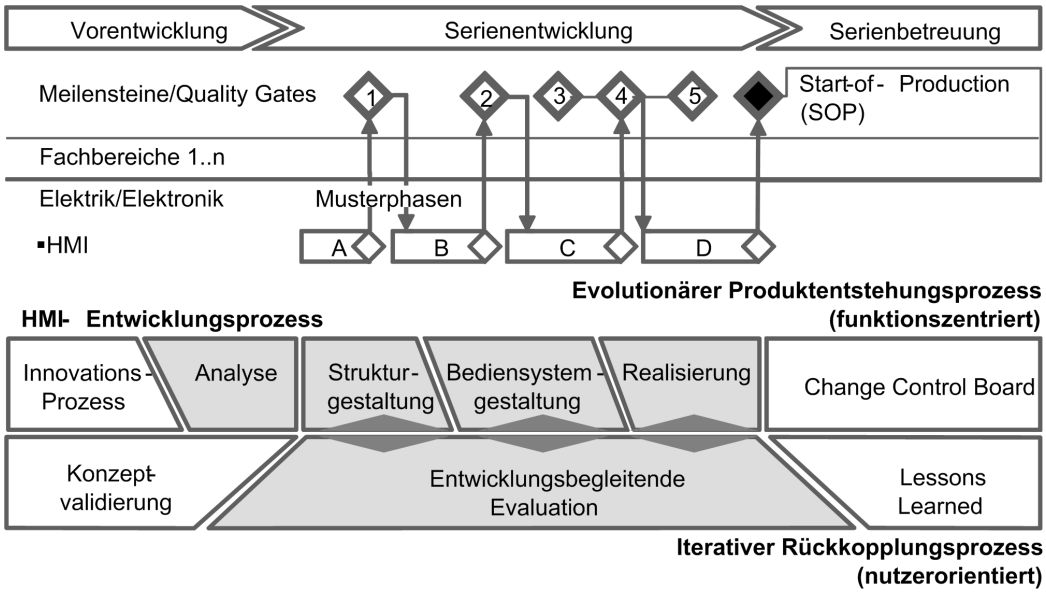

Abbildung 5.1: HMI-Entwicklungsprozess der Porsche AG (Bock 2007)

In diesen drei Entwicklungsstufen sind die Fachbereiche *Antrieb*, *Fahrwerk*, *Karosserie* und *Elektrik/Elektronik* beteiligt. Der zuletzt genannte Fachbereich ist für die Entwicklung der HMI-Komponenten verantwortlich. Alle Fachbereiche werden durch sogenannte *Quality Gates* synchronisiert, d.h. die Ergebnisse aus den verschiedenen Fachbereichen werden zusammengeführt, um so den Gesamtkontext der einzelnen Systeme zu erhalten. Diese Marken sind mit Meilensteinen aus dem Projektmanagement vergleichbar. Meistens findet im Anschluss an ein sogenanntes *Quality Gate* eine Abnahme durch die verantwortlichen Abteilungen statt. Die Probandentests finden meistens unter einem Codenamen an abgelegenen Orten der Welt statt. Personen aus dem höheren Management bewerten in Zusammenarbeit mit den Ingenieuren die entwickelten Prototypen. Wurden alle Musterphasen und *Qualitiy Gates* positiv abgeschlossen, so beginnt die nächste Phase, der *Start-of-Production* (SOP). Ab diesem Zeitpunkt läuft die Serienproduktion der Fahrzeuge an. Diese gehen auf direktem Weg zum Kunden, denn die Porsche AG produziert ausschließlich auf Anfrage.
In der ersten Entwicklungsphase, der *Vorentwicklung*, werden grundlegende Forschungsschwerpunkte im Umfeld der Fahrzeuge analysiert. So werden neue technologische Produkte von der Abteilung eingehend untersucht und bewertet, um so zukünftige Konzepte bereits früh in den Entwicklungsprozess aufzunehmen. Ziel dabei ist es, Produkt- und Prozessinnovationen abzusichern, um so den Aspekt der Fahrzeugsicherheit bei der Entwicklung zu gewährleisten. Dieser ist bei der Fahrzeugentwicklung von großer Bedeutung. Der Innovationsprozess der Porsche AG, der sich in eine Konzept- und eine Absicherungsphase einteilen lässt, steuert die systematische Weiterentwicklung von neuen Konzepten im Fahrzeug.
In der zweiten Phase der *Serienentwicklung* werden in einem evolutionären Prozess die einzelnen Komponenten der Fachbereiche in Zusammenarbeit mit Zulieferern entwickelt und ausgearbeitet (Bock 2007). In dieser Phase folgt auf die Spezifizierung des Auftraggebers, also der Porsche AG, das Gespräch mit potentiellen Lieferanten. Nach der Vergabe des Auftrags erstellt der Auftragnehmer ein Pflichtenheft, welches von der Porsche AG nach Überprüfung aufgenommen wird. Nach der Abnahme wird in sehr enger Kooperation mit dem

Lieferanten das Produkt für die anschließende Serienfertigung realisiert. Mit dem Produktionsstart *(Start-of-Production)* beginnt die *Serienbetreuungsphase*. In dieser Phase werden eventuelle Änderungen, wie beispielsweise Defekte von elektronischen Bauteilen, im sogenannten *Change Control Board* festgehalten. Ein Grund hierfür könnte sein, dass ein Kunde oder ein Zulieferer einen Mangel erkennt.

Im Folgenden wird nun der HMI-Entwicklungsprozess als Teil des gesamten Entwicklungsprozesses näher erläutert. In diesem Prozess werden neben dem Porsche Fahrerinformationssystem (PCM) beispielsweise auch das Kombiinstrument und die Klimabedienung entwickelt. Die Abteilungen, die an der Befragung teilnahmen, gehörten zum einen zur *Vorentwicklung (Abteilung Elektrik/Elektronik Vorentwicklung)* sowie zur *Serienentwicklung (Abteilung Elektrik /Elektronik Interieur)*. Die letztgenannte Abteilung nahm auch an der *Serienbetreuung* teil, um beispielsweise mit dem Zulieferer Software-Updates auszuarbeiten. Die Updates wurden anschließend an die Porsche-Händler weitergegeben. Somit nahmen an der Analyse des Porsche HMI-Entwicklungsprozesses Entwickler teil, die von der *Vorentwicklung* bis zur *Serienbetreuung* am Prozess beteiligt sind (vgl. Kapitel 2). Im Rahmen dieser Arbeit wird nur kurz auf die Phasen der *Analyse*, *Strukturgestaltung*, *Bediensystemgestaltung* und *Realisierung* eingegangen. Eine ausführliche Beschreibung der Phasen nimmt Bock (2007) in seiner Arbeit vor.
In den ersten Phasen der *Analyse* werden die Anforderungen an das System festgelegt. Hierbei fließen die Erkenntnisse zur Gestaltung von Mensch-Maschine-Schnittstellen aus dem Innovationsprozess (vgl. Abbildung 5.1) mit ein. Des Weiteren erfolgt in dieser Phase die Analyse der Nutzer sowie deren Anforderungen. So wird beispielsweise im Rahmen einer Benutzungshäufigkeitsanalyse (BHA) das aktuelle Fahrerinformationssystem evaluiert, um so herauszufinden, was von den potentiellen Nutzern an Funktionen und Optionen benutzt wird. Bei der Analyse lassen sich auch die Navigationszeiten der Probanden aufzeichnen (Schwarz 2005). Darüber hinaus werden durch ständige Marktbeobachtungen die aktuellen Techniktrends ermittelt. Nach Bock (2007) wird während der Analysephase ein mentales Modell der Nutzer abgeleitet, das die Technologieorientierung, Nutzungsorientierung und Handlungsorientierung beinhaltet.
Aufbauend auf den definierten Anforderungen und Aufgaben folgt nun die *Strukturgestaltungsphase*. In dieser Phase wird neben dem Plattformkonzept ein sogenanntes plattformunabhängiges Benutzungsmodell abgeleitet (Bock 2007). Ohne die spätere Hard- oder Softwareplattform zu kennen, beschreibt das Benutzungsmodell das Zusammenspiel von Nutzer und System. Ferner werden in dieser Phase ein Technikkonzept, ein Bediensystemkonzept und ein Anzeigekonzept von den Entwicklern abgeleitet. Im Technikkonzept wird beispielsweise von *technischen Experten* der Bauraum für das Fahrerinformationssystem definiert. Im Bedienkonzept erarbeiten die *Usability-Engineers* in enger Abstimmung mit den *technischen Experten* erste Menüstrukturen für die Dialoge. Im Anzeigekonzept wird ein Groblayout für die Inhalte der Bildschirme konzipiert, d. h. im Anzeigekonzept geht es um ein erstes Layout für die Inhalte der Bildschirme und deren Aufteilung. Mit diesem Punkt beschäftigt sich der *Designer*.
Im Rahmen der *Bediensystemgestaltung* wird das *Interaktions-* und *Oberflächendesign* genau

spezifiziert, so geht es hierbei um die Auswahl geeigneter Hard- und Softwareplattformen. In dieser Phase wird zwischen der Grob- und Detailgestaltung unterschieden. Die Ergebnisse der Analyse- und Strukturgestaltungsphase fließen in die Bediensystemgestaltung mit ein. In der Grobgestaltung des Bedienkonzepts werden beispielsweise Elemente für die Interaktion zwischen Nutzer und Bediensystem festgelegt. Im Anzeigekonzept der Gestaltungsphase wird zum Beispiel das Blendendesign der Hardware definiert. Darüber hinaus wird bei der Grobgestaltung im Dialogkonzept eine grundlegende Menüstruktur erarbeitet. In der Detailgestaltungsphase werden die zuvor vorgestellten Konzepte optimiert, so wird beispielsweise im Bedienkonzept das genaue Scrollverhalten bei der Eingabe spezifiziert. Gleiches gilt für die anderen Phasen: Zum einen wird im Anzeigekonzept des Designers das genaue Aussehen der einzelnen Bildschirme definiert, zum andern vom *Usability-Engineer* eine detaillierte Menüstruktur ausgearbeitet.
Eine *Feinabstimmungsphase* verbindet die Bediensystemgestaltung mit der *Realisierung*. Hierbei haben die Ingenieure ständigen Kontakt zu den Zulieferern, um so eine mögliche Prozessinkonsistenz sofort zu beseitigen. Im Rahmen der Realisierung werden umfangreiche Tests in den Fahrzeugen durchgeführt, um so die realen Bedienungen bei Erprobungsfahrten zu testen. So bildet nicht die Realisierung den Schwerpunkt der Phase, sondern die funktionalen Tests der Systeme. Die Realisierung wird parallel zu den Musterphasen von Zulieferern und Herstellern getätigt (Bock 2007).

5.2 Unterstützungsmöglichkeit in den Prozessphasen

Um die Unterstützungsmöglichkeiten im HMI-Entwicklungsprozess der Porsche AG einordnen zu können, wird dieser in die Phasen *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung* aufgeteilt (vgl. Abschnitt 5.1). Dabei werden neben den organisatorischen, wie beispielsweise Unterstützung der verschiedenen Rollen (vgl. Kapitel 2), auch die technischen Gesichtspunkte eingehend erläutert, d.h. an welcher Stelle im Entwicklungsprozess die beteiligten Personen unterstützt werden können, um so ein detailliertes Verständnis für die Anforderungen an die Konzeption der Unterstützungskomponente zu erlangen.
Die folgende Abbildung 5.2 zeigt die Aufteilung der Phasen zusammen mit den am Entwicklungsprozess beteiligten Personen.

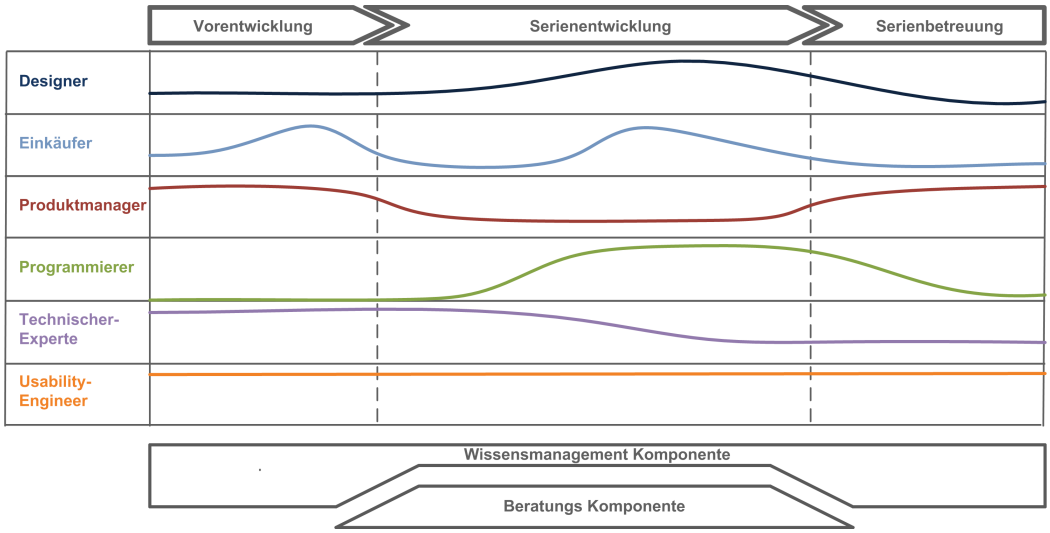

Abbildung 5.2: Projektphasen und Rollen

Darüber hinaus kennzeichnen die eingefärbten Linien im Schaubild, wann und in welchem Umfang die entsprechenden Personen in der jeweiligen Phase beteiligt sind.
So ist zum Beispiel der *Usability-Engineer* während der kompletten Projektphase am HMI-Entwicklungsprozess beteiligt. Der *Programmierer* dagegen ist ausschließlich in der Phase der *Serienentwicklung* und *Serienbetreuung* tätig. Der *Produktmanager* ist an der *Vorentwicklung* und an der *Serienbetreuung* beteiligt: Er führt in der ersten und dritten Phase des HMI-Entwicklungsprozesses Markt- und Techniktrenduntersuchungen durch. Der *Einkäufer* wird hinzugezogen, wenn ein Bauteil, wie zum Beispiel ein Foliendisplay für einen Touchscreen, in der *Vorentwicklung* benötigt wird, um neue Tests durchzuführen. Der *Einkäufer* ist analog in der Phase der *Serienentwicklung* mit eingebunden. Dabei geht es meistens um Preisverhandlungen mit den Zulieferern, die in der engeren Auswahl für die Beschaffung neuer Komponenten für das Bediensystem stehen. Der *Designer* kommt primär im Rahmen der *Serienentwicklung* zum Einsatz, wenn es um die Erstellung des Blendendesigns geht. Zum einen entwickelt er in der Grobgestaltungsphase das Masken- und Blendendesign, zum anderen spezifiziert er in der Detailgestaltungsphase beispielsweise die Schrift- und Farbabstimmung des Bediensystems. Die Wissenskomponente wird hauptsächlich in den Phasen der *Vorentwicklung* und *Serienbetreuung* benötigt (vgl. Abbildung 5.2). So können schon während der *Vorentwicklung* Aktivitäten wie beispielsweise Kosten- und Ressourcenplanungen

in einer Art Qualitätsplan den einzelnen Rollen frei zugänglich gemacht werden. Darüber hinaus werden über die Nutzung von bisherigen Erfahrungen und qualitativ hochwertigen Literaturquellen die Projektergebnisse verbessert.
Im Bereich der *Serienentwicklung* kommt die Beratungskomponente zum Einsatz (vgl. Abbildung 5.2), die die Entwickler während der Erstellung interaktiver Prototypen unterstützt. In der *Serienentwicklung* kann jederzeit auf die Wissensdatenbank zugegriffen werden. Außerdem können genaue Beschreibungen des Arbeitsplatzes der Rollen in der Wissensdatenbank abgelegt werden. So kann das Wissen im Unternehmen gehalten werden, falls ein Mitarbeiter das Unternehmen verlässt (vgl. Abschnitt 3.4).
Im folgenden Abschnitt wird auf die einzelnen Phasen näher eingegangen, indem beschrieben wird, wie die einzelnen Rollen genau unterstützt werden können. Das Herzstück, die *Serienentwicklung*, wird in die Dimensionen *Oberflächen-*, *Interaktions-* und *Informationsdesign* aufgeteilt (vgl. Kapitel 2), um so detailliert die Unterstützungsfunktionen herausarbeiten zu können.

5.2.1 Vorentwicklung

In der *Vorentwicklung* liegt das Hauptaugenmerk auf der *Analysephase*. Der *technische Experte* und der *Usability-Engineer* können sehr stark von Wissenskomponenten profitieren, indem sie die bisher gemachten Erfahrungen *(Lessons Learned)*, welche sie aus anderen Projekten gesammelt haben, abrufen können. Die Erfahrungsberichte über den Einsatz der Methoden und Werkzeuge können zu jedem Projekt abgespeichert werden. Zusätzlich haben sie einen sehr schnellen Zugriff auf Normen, Standards und Guidelines. Der *Produktmanager* kann die Beobachtungen der Techniktrends in der Wissensdatenbank ablegen, um so allen Mitarbeitern aktuelle Informationen frei zugänglich zu machen. Die *Einkäufer* können die bisherigen Angebote von früheren Projekten ablegen und somit eine Preistransparenz schaffen. So können die verschiedenen *Einkäufer* je nach Bauteil die eingegebenen Preise vergleichen und entsprechend die preisgünstigsten Zulieferer auswählen.
Darüber hinaus kann eine Art Produkt-Styleguide eingeführt werden (vgl. Kapitel 3.1.1). In dieses standardisierte Dokument tragen alle Personen, die am HMI-Entwicklungsprozess beteiligt sind, ihre erzielten Ergebnisse ein. Dieses Dokument liegt frei zugänglich auf der Wissensdatenbank. In dem Dokument wird genau der aktuelle Stand der derzeitigen Entwicklung beschrieben. Des Weiteren beinhaltet das Dokument neben einer Beschreibung der Ergebnisse auch den Autor, um gegebenenfalls bei eventuellen Rückfragen einen Ansprechpartner zu haben. Dieses Dokument wird über die Prozessphasen der *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung* hinaus immer auf aktuellem Stand gehalten, um den Entwicklungsprozess transparent zu machen.

5.2.2 Serienentwicklung

Die *Serienentwicklung* ist der Schwerpunkt des HMI-Entwicklungsprozesses. Darüber hinaus arbeiten die *Usability-Engineers*, die *technischen Experten*, die *Designer*, die *Einkäufer* und die *Programmierer* in dieser mittleren Phase gemeinsam an der Gestaltung des Bediensystems. Im Folgenden werden die einzelnen Dimensionen, wie das *Oberflächen-*, das

Informations- und das *Interaktionsdesign*, eingehend erläutert. Zusätzlich werden in den folgenden Abschnitten die Unterstützungsmöglichkeiten im Hinblick auf die oben genannten Rollen erklärt.

Oberflächendesign

Die Möglichkeit, die grafische Darstellung in einem GUI-Builder zu ändern, führt zu einer bedeutenden Beschleunigung im HMI-Entwicklungsprozess. Darüber hinaus können mithilfe des GUI-Builders die Designer ohne Programmierkenntnisse direkt in den Prozess eingreifen. Die *Designer* können sich auf ihre Domäne (Design) konzentrieren, ihre Kreativität ausleben und werden nicht durch die technische Umsetzung gehemmt.
Nach Reiterer (2000) ist für die Entwicklung von Benutzeroberflächen Folgendes von Bedeutung: *"to discover helpful, unobstrusive, structured, and organized ways to integrate the use of principles, guidelines, standards, style guides, and design rules into the design process without stifling creativity."* (Seite 3)
So müssen Verfahren gefunden werden, die die relevanten Maße zur ergonomischen Gestaltung von Benutzeroberflächen direkt als Gestaltungshinweise innerhalb des Entwicklungsprozesses der Porsche AG umsetzen.
Für die Unterstützung der *Designer* im Bereich des *Oberflächendesigns* sollen Daten zur Lage und Anordnung der Oberflächenelemente zur Verfügung gestellt werden, d.h. die Kriterien beschreiben die Position der Elemente auf dem Bildschirm, um so beispielsweise die horizontale und vertikale Balance[1] der Oberfläche zu errechnen. Darüber hinaus soll das Werkzeug die Konsistenz der Benutzeroberfläche prüfen. Die konsistente Gestaltung von Benutzeroberflächen ist eines der wichtigsten Kriterien im Bereich der Gestaltung von interaktiven Systemen (vgl. Abschnitt 3.2.2). Hierbei können beispielsweise Konsistenzlinien hilfreich sein, um die Menüeinträge der PCM-Bildschirme auszurichten. Die richtige Farbauswahl ist für Fahrerinformationssysteme von besonderer Bedeutung. Hierbei muss beispielsweise die Sonneneinstrahlung berücksichtigt werden. So kann dem Entwickler dadurch geholfen werden, dass der spektrale Abstand von Farben gemessen wird. Dieser spektrale Farbabstand kann dann verwendet werden, um zum Beispiel die Lesbarkeit der Bildschirmelemente zu erhöhen.
Es kann durchaus passieren, dass der *Usability-Engineer* selbst leichte Änderungen am *Oberflächendesign* vornehmen will. Dies ist beispielsweise der Fall, wenn er die Anzahl der Menüeinträge erhöhen will, um sein neu erarbeitetes Bedienkonzept zu testen. Hier ist es sinnvoll, geprüfte ergonomische Bildschirme in einer Bibliothek abzuspeichern. Diese ergonomisch geprüften Bausteine kann der *Usability-Engineer* als Vorlage für weitere Entwicklungsschritte nutzen. Die Wissenskomponente kann hier ebenfalls für die Identifizierung und Recherche von Entwurfsentscheidungen und Entwurfsbegründungen herangezogen werden. Darüber hinaus kann eine Wissensdatenbank die Normen und Standards bereitstellen, die bei der Erstellung von Benutzeroberflächen für Fahrerinformationssysteme von besonderer Relevanz sind (vgl. Abschnitt 3.2.2). Ferner haben die Entwickler einen schnellen Zugriff auf den Porsche Style-Guide, wenn dieser in der Wissensdatenbank abgelegt ist.

[1] Maß für Ausgewogenheit einer Benutzeroberfläche.

Informationsdesign

Im Hinblick auf das *Informationsdesign* ist das Ziel, die Entwickler im Bereich der Darstellung von textuellen Informationen zu unterstützen. Für die Erlernbarkeit von interaktiven Systemen ist es besonders wichtig, dass die Begrifflichkeiten bei den unterschiedlichen Dialogfolgen die gleiche Schreibweise haben, um so Missverständnisse zu vermeiden. In Johnson (2000) werden bezeichnende textuelle Mängel umfangreich vorgestellt. Es folgt nun eine kurze Beschreibung möglicher unterstützender Funktionalitäten.
An der Spezifikation von Funktionen arbeiten verschiedene *technische Experten* zusammen. So könnte eine Funktion prüfen, ob jeder Bildschirm mit einem Pfad (zum Beispiel Main>Werkseinstellung) versehen ist, um eine eindeutige Kennzeichnung der Bildschirme zu erlangen. Das System alarmiert beispielsweise *on-the-fly* den Entwickler während der Eingabe der Texte. Zusätzlich kann mithilfe der Ontologie eine Hierarchie für Begriffe eingeführt werden. So dürfen beispielsweise nur spezielle Oberbegriffe in der oberen Menüleiste vorkommen. Die Begrifflichkeiten einer Domäne können in Menübäumen angelegt werden. Hier wird mittels der Ontologie entsprechend geprüft, ob die modellierte Menüstruktur mit den Begriffen aus dem Baum übereinstimmt. Die Verwaltung der Begrifflichkeiten kann mithilfe eines Textmanagementprogrammes erfolgen. Darüber hinaus ist es denkbar, dass das Programm auch beispielsweise Übersetzungen der Menüeinträge in verschiedenen Sprachen anbietet.
Durch eine Rechtschreibprüfung kann dieses Feature erweitert werden. Unterläuft dem *technischen Experten* ein Rechtschreibfehler, so wird dieser wie bei einer herkömmlichen Office Anwendung darauf aufmerksam gemacht.
Weiter gibt es die Möglichkeit, dass ein Wortergänzungsverfahren, wie es beispielsweise in Microsoft *Excel* implementiert wurde, die Entwickler unterstützt. So kann die textuelle Konsistenz der verwendeten Begriffe stark verbessert werden. Diese könnten zusätzlich die Funktionen in verschiedenen Landessprachen anzeigen, die aus dem Textmanagementprogramm stammen.
Die *Usability-Engineers* werden zusätzlich noch unterstützt, indem Bereiche von den Designern definiert werden können, in denen nur eine bestimmte Schriftgröße zulässig ist. Nach der ISO 15008 darf die Schriftgröße bei Fahrerinformationssystemen die 10 Punkte Marke nicht unterschreiten. Sonst ist die Ablenkung während der Fahrt zu groß.
Weiter ist es möglich, für die Menüeinträge eine Längenprüfung zu implementieren, da eine bestimmte Wortlänge nicht überschritten werden sollte. Gibt der Benutzter einen zu langen Menüeintrag ein, so wird er vom System in Form einer Anzeige darauf hingewiesen, dass das eingegebene Wort zu lang ist. Hierbei müssen auch die verschiedenen Landessprachen in Betracht gezogen werden. Ein Wörterbuch kann dabei Abhilfe schaffen.

Interaktionsdesign

Im *Interaktionsdesign* werden neben der Entwicklung der Menüstruktur geeignete Bedienelemente für die Ein- und Ausgabe definiert, um so eine konsistente und erwartungskonforme Bedienung gewährleisten zu können (Normen 1988). Es werden nun ebenfalls mögliche Unterstützungen kurz vorgestellt.
Mithilfe des *Interaktionsdesigns* kann das Bedienkonzept geprüft werden, um so Aufschlüsse über eine konsistente Menüstruktur zu erhalten. Da das Fahrerinformationssystem meistens während der Fahrt (als Sekundäraufgabe) bedient wird, muss ein gut strukturiertes Bedienkonzept vorhanden sein. So gilt es beispielsweise zu prüfen, ob eine breite oder tiefe Menüstruktur für den Kunden besser zu bedienen ist.
In diesem Zusammenhang beauftragte 2004 die Bundesanstalt für Straßenwesen das interdisziplinäre Zentrum für Verkehrswissenschaften der Universität Würzburg. Das Projekt lief unter dem Namen: *"Kompetenzerwerb für Fahrerinformationssysteme: Bedeutung von Bedienkontext und Menüstruktur"* (Rauch und Totzke 2004). Mithilfe von 24 Probanden wurden zwei Menüsysteme in einer Dual-Task-Situation untersucht. Eine breite Struktur mit je acht Optionen auf zwei Menüebenen (8x8 Struktur) und eine tiefe Menüstruktur mit je zwei bzw. vier Optionen pro Ebene (4x2x2x4) wurde herangezogen. Beide Menüs bestanden aus 64 Funktionen. Die Versuchspersonen mussten 48 Aufgaben während der Fahrt bearbeiten. Dabei wurden die *mittlere Navigationszeit* (Dauer, in der die Probanden innerhalb des Menüsystems navigieren), *mittlere Schrittdauer* (Dauer pro Einzelschritt im Menü, berechnet aus *mittlerer Navigationszeit* der Schritte) und überflüssige Wegstrecke (Parameter der Bediengüte aus dem Verhältnis tatsächlich gegangener Schritte zu notwendigen Schritten für die Bearbeitung einer Aufgabe) betrachtet.
Während des Versuches zeigten sich vor allem zu Lernbeginn Einbußen in der Bediengeschwindigkeit durch Interferenz mit der parallel zu bearbeitenden Aufgabe. Die Aufgabe wurde bei breiten Menüs schlechter gelöst. Ein breites Menü benötigt eine längere Blickdauer bei einer vergleichbaren Blickanzahl zur Bedienung als ein tiefes Menü. Es wurde deutlich: Je kürzer und gleichmäßiger die notwendigen Blicke zur Bedienung sind, desto leichter werden die Aufgaben erledigt. Daraus lässt sich ableiten: Tiefe Menüs sind im Fahrzeug den breiten überlegen (Rauch und Totzke 2004).
Ferner können im Bereich des *Interaktionsdesigns* die *Usability-Engineers* unterstützt werden, indem das System sie auf sogenannte *Deadlocks* hinweist, d.h. das System löst Alarm aus, wenn beispielsweise bei der Ausarbeitung der Menüstruktur keine Rückkehr (Negativpfad) aus den PCM-Bildschirmen möglich ist. Dies ist der Fall, wenn ein Entwickler die PCM-Bildschirme mit dem falschen Bedienelement verbunden oder diese sogar völlig vergessen hat.
Darüber hinaus sollte auch vom System vorgegeben werden, mit welcher Art von Bedienelementen die Bildschirme verbunden werden dürfen. Beim PCM 2.1 unterscheiden sich beispielsweise die Bedienelemente in *Dreh-Druck-Steller* (rechts und links), *Hauptfunktionstasten*, *Nebenbedientasten*, *Zifferntasten* und *Telefonbedientasten* (vgl. Abbildung 5.3).

Abbildung 5.3: Bedienelemente PCM 2.1

5.2.3 Serienbetreuung

In der *Serienbetreuung* können die anfallenden Informationen im Wissensmanagementsystem abgelegt werden. Diese können dann für die *Vorentwicklung* als *Lessons Learned* wieder verwendet werden.
Darüber hinaus werden die auftretenden Fehler in der *Serienbetreuung* systematisch bei der Produktverbesserung berücksichtigt, indem diese auf dem *Change Control Board* abgelegt werden. Dieses Board kann in das Wissensmanagementsystem integriert werden.
Ferner können die Erfahrungen in der Wissensdatenbank abgelegt werden, die beispielsweise direkt von den Vertragshändler der Porsche AG kommen, denn diese haben täglich mit den Porsche Kunden zu tun. So können Anregungen für die neuen Produkte entstehen. Weiter könnten die Porsche Vertragshändler direkter über das Wissensmanagementsystem vernetzt werden und bei eventuell auftretenden Mängeln mit dem verantwortlichen Entwickler aus der *Serienbetreuung* Kontakt aufnehmen. So ist eine schnelle Diagnose durchführbar, da die Problemstellung strukturiert aufgenommen werden kann.

5.2.4 Projektübergreifende Unterstützung

Das projektübergreifende Wissen dient zur Verbesserung der Qualität von Prozessen. Diese wirken sich auch unmittelbar auf die Produkte aus, d.h. erhöht sich die Prozessqualität, so wirkt sich diese auch auf die Produktqualität aus. Darüber hinaus werden die gesammelten Erfahrungen in den Projekten als Anregung aufgefasst und für die Entwicklungsprozesse genutzt, um die Wissensbasis an die neuen Erfordernisse anzupassen. Entdeckt beispielsweise ein Entwickler ein Verbesserungspotenzial im Prozess, so muss diese Anregung in die Wissensbasis mit aufgenommen werden, denn keiner kennt die einzelnen Arbeitsschritte besser als der Entwickler selbst. Frieling und Reuther (1993) nennen diesen Prozess der Verbessrung über Projektgrenzen hinweg *Lernende Organisation*. Durch diese Maßnahmen der ständigen Verbesserung erhöht sich die Prozessqualität. Nach Capurro (1998) lassen sich die Begriffe *Lernende Organisation*, *Informationsmanagement* und *Wissensmanagement* nicht trennen: *"Neues Wissen in Organisationen ist das Ergebnis von Lernprozessen. Lernen basiert wie-*

derum sowohl auf dem gegenseitigen Austausch von expliziten Informationen als auch, auf der Wechselwirkung zwischen explizitem und implizitem Wissen." (Seite 3)

5.3 Ableitung des Konzepts für die Beratungskomponente und das Wissensmanagementsystem

Die konzeptionelle Aufteilung der Unterstützungsfunktionen erfolgt nach der Analyse des in Abschnitt 5.1 vorgestellten HMI-Entwicklungsprozesses in zwei Module. Zum einem soll die Beratungskomponente in den GUI-Builder und in das CASE-Tool (MetaEdit+) integriert werden (Heilig 2007), zum anderen soll die Wissensdatenbank in einer eigenständigen Applikation realisiert werden. Auch Balzert (1998) klassifiziert die Unterstützungssysteme in Beratungs- und Auskunftssysteme (vgl. Abschnitt 3.3.2). Die Trennung der beiden Module macht die Bearbeitung über Abteilungs- und Unternehmensgrenzen hinweg möglich. Dadurch ist es nicht notwendig, dass alle Mitarbeiter im Unternehmen die Werkzeugkette an ihrem Arbeitsplatz installiert haben. Vielmehr können sie das Wissensmanagementsystem dahin gehend nutzen, eine grundsätzliche Beratung beispielsweise in den Themengebieten Usability-Normen und Standards durchzuführen.
Mithilfe der Beratungskomponente werden die Entwickler in der *Serienentwicklung* unterstützt. Darüber hinaus muss jedoch gewährleistet sein, dass die Entwickler aus den Werkzeugen der Werkzeugkette auf das Wissensmanagementsystem zugreifen können, um die benötigten Informationen zu erhalten. Ein weiterer Punkt, der für die Trennung der Systeme spricht, ist die daraus resultierende Erleichterung der Bedienung, weil die Systeme völlig unabhängig voneinander in den Entwicklungsprozess eingegliedert werden können. So können schon in sehr frühen Phasen die *Produktmanager* die Marktbeobachtungen der Bediensysteme in die Wissensdatenbank einpflegen. Durch die konzeptionelle Aufteilung der Module benötigt der *Produktmanager* die Werkzeugkette nicht an seinem Arbeitsplatz. Er hat aber die Möglichkeit, durch das Wissensmanagementsystem seine Entwickler zu erreichen. Darüber hinaus könnte so das Wissensmanagementsystem über Abteilungsgrenzen hinweg im Porsche Konzern eingesetzt werden.
Der Name der Unterstützungskomponente für den Porsche HMI-Entwicklungsprozess lautet *PoCKET*. *PoCKET* steht für **Po**rsche **C**onsulting and **K**nowledege **E**ngineering **T**ool. Die folgende Abbildung 5.4 soll die Aufteilung des Systems verdeutlichen.

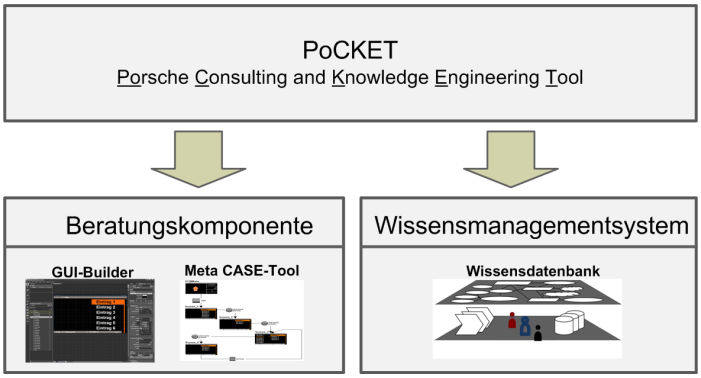

Abbildung 5.4: Aufteilung der Komponenten

5.4 Konzept der Beratungskomponente

In diesem Abschnitt wird das Konzept der Beratungskomponente vorgestellt. Die Komponente wird vornehmlich in der *Serienentwicklung* eingesetzt.
Um die Unterstützungskomponente für die Werkzeugkette entwerfen zu können, müssen die Informationen der Komponenten *Oberflächendesign*, *Informationsdesign* und *Interaktionsdesign* maschinenlesbar vorliegen, weil sonst kein Regelwissen, wie beispielsweise Abstand der Farbgebung angewendet werden kann. Anschließend wird die abgeleitete Struktur der Beratungskomponente eingehend erläutert. Darüber hinaus wird die Integration des im Rahmen dieser Arbeit entworfen Konzepts in die Werkzeugkette beschrieben.

5.4.1 Formalisierbarkeit der Dimensionen

Bei der heutigen Entwicklung von interaktiven Systemen gilt das Gestalten von Benutzeroberflächen als eine intuitive Disziplin, in der sehr viel Eingebung und Erfahrung notwendig sind. Trotzdem befassten sich einige in der Vergangenheit mit der Ableitung von Designkriterien, um die Benutzeroberflächen mit einem Maß formalisierbar zu machen. Nach Sears (1995) können Metriken definiert werden, welche die Operationalisierung von Gestaltungseigenschaften darstellen können. So wurden schon (1995) durch Sears sogenannte *User-Interface-Metriken* definiert, die ästhetische und technische Dimensionen einer Benutzeroberfläche beschreiben. Ziel dabei ist es, dass die Entwickler die Eigenschaften von Benutzeroberflächen automatisch evaluieren können (vgl. Kapitel 4). Auch Mahajan, Shneiderman (1997) und Ivory (2001) haben einige Maße zur Prüfung der Anordnung und Lage der Oberflächenkomponenten aus Normen und Guidelines abgeleitet. Im Bereich des *Oberflächendesigns* lassen sich im Gegensatz zum *Informations-* und *Interaktionsdesign* mehrere Metriken ableiten. Nach Farenc (1997) sind jedoch der automatischen Evaluierung gewisse Grenzen gesetzt. Er führte eine Untersuchung mit dem Werkzeug *ErgoVal* (vgl. Abschnitt 4.2.1) durch und fand heraus, dass nur einzelne Teile der ergonomischen Guidelines des Designs operationalisiert werden können (Farenc 1997). Dies führte er auf die fehlenden Informationen der Benutzer zurück (Farenc 1997).
Im Folgenden werden einige Metriken beschrieben, die in die Dimensionen *Oberflächen-*, *Interaktions-* und *Informationsdesign* aufgegliedert sind.

Oberflächendesign

Um eine automatische Prüfung der Benutzeroberflächen realisieren zu können, muss ein Beschreibungsformat von Oberflächen erarbeitet werden. Die bekannten Beschreibungsformalismen befassen sich meist mit der Darstellung von Webseiten (vgl. Kapitel 4).
Bei der Dialogbeschreibung haben sich die objektorientierten Ansätze durchgesetzt. Neben Positionen wie die X- und Y- Position sowie die Höhe und Breite können zusätzlich weitere Attribute angegeben werden (Hamacher 2006). Die Oberflächenelemente im Dialog müssen formal durch Metriken beschrieben werden (Ivory 2001). Derartige Maßzahlen müssen auch für die Prüfung der Farbgebung gefunden werden. In diesem Abschnitt der Arbeit werden einige Ansätze geometrischer Gestaltungsmaße, sowie Maße für die Farbgebung vorgestellt. Ferner muss bei der Oberflächengestaltung von Dialogsystemen verstärkt auf die Gleichför-

migkeit der Anordnung von Elementen geachtet werden (Shneiderman und Plaisant 2005). Dies gilt in allen Bereichen der Systemgestaltung. Zudem muss während der Bedienung des interaktiven Systems die Dialogfolge konsistent sein (vgl. Abschnitt 3.2.2). Die effiziente Dialoggestaltung wirkt sich vor allem auf die bessere Erlernbarkeit aus *(ease-of-learn)* und somit auf die Zufriedenheit des Endkunden. Nach Shneiderman und Plaisant (2005) kann ein durchweg konsistent gestaltetes System eine bis zu 25% schnellere Erledigung der Dialogaufgabe mit sich führen. Die Oberflächenkonsistenz ist sehr schwer formalisierbar bzw. messbar. Es müssen dafür die Gestaltungsmaße berücksichtigt werden. Zu diesen zählen die *Positionierung* der Elemente, sowie deren *Größe*, *Anordnung* und *Farbgebung*. Bei der Prüfung der Oberflächenkonsistenz müssen die unterschiedlichen Werte der Dialoge miteinander verglichen werden. So lassen sich die Veränderungen der Gestaltung beobachten. Darüber hinaus muss zur automatischen Überprüfung der Oberfläche ein Verfahren entwickelt werden, um die Elemente selektieren zu können, sodass diese, wie oben beschrieben, mit Metriken formalisiert werden können.
Beirekdar et al. (2002) stellt in seinem Konzept (vgl. Abschnitt 4.2.4) einen Ansatz vor, wie die entsprechenden Elemente ausgewählt und anschließend geprüft werden können. Basierend auf dem Ansatz wird für diese Arbeit ein ähnliches Konzept angewendet.
Zu Beginn sollen die Oberflächenelemente ausgewählt werden (vgl. Abbildung 5.5), deren Attribute überprüft werden sollen, d.h. bei der Prüfung der Farbgebung wird beispielsweise das Attribut *Background=FFFFFF*[2] identifiziert. Im nächsten Schritt müssen die Elemente in Beziehung zueinander gebracht werden, z.B. eine Form vor einem Hintergrund *Rectangle Fill=FF0000*[3]. Jetzt werden diese beiden Elemente, Hintergrundfarbe und Farbe des Rechtecks, zu einer atomaren Gruppe zusammengefasst (Beirekdar et al. 2002). Zuletzt werden die Metriken, die in einer Regelbasis abgelegt sind, auf die gruppierten Attribute angewendet.

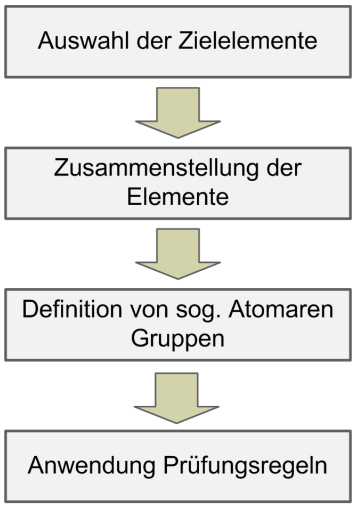

Abbildung 5.5: Überprüfung der Oberfläche (Beirekdar et al. 2002)

Im Folgenden werden geometrische Kennzahlen bzw. Metriken vorgestellt, mit deren Hilfe Benutzeroberflächen formalisiert werden können.

[2]Hexadezimale Farbdefinition von Weiß.
[3]Hexadezimale Farbdefinition von Rot.

Display Aspect Ratio Das Display Aspect Ratio (DA) beschreibt das Verhältnis zwischen der Breite (D^h) und Höhe (D^b) des verwendeten Bildschirms. Der optimale Wert liegt zwischen 0,5 und 0,8 (Mahajan und shneiderman 1997).

$$DA = \frac{D^{[h]}}{D^{[b]}} \tag{5.1}$$

Anzahl der Displayelemente Dieser Wert wird zur Berechnung der Oberflächenkonsistenz verwendet. Er gibt die Anzahl (A_E) der dargestellten Elemente (E) auf dem Bildschirm an. Das System zählt alle Elemente auf dem Bildschirm. So kann beispielsweise der Entwickler eine Grenze für die maximale Anzahl der Elemente definieren, um die Benutzeroberfläche nicht zu überladen (Mahajan und shneiderman 1997).

$$A_E = |E| \tag{5.2}$$

Abstand der Randpunkte Das Maß beschreibt den Abstand (Sr) zwischen dem äußersten Bildschirmelement ($d_i^{[x]}$) und dem Rand ($d_j^{[x]}$) des Bildschirms (D). Dieser Wert kann auch für die Bestimmung der Oberflächenkonsistenz herangezogen werden. Das folgende Beispiel zeigt die Berechnung des Abstandes der Pixel zum rechten Rand (Mahajan und shneiderman 1997). Ferner wird ein Mindestabstand der Randpunkte in der ISO 15008 für Anzeigen im Fahrzeug vorgeschrieben. Hamacher (2006) gibt zur Berechnung folgende Formel an:

$$Sr = d_i^{[x]} - d_j^{[x]} \qquad \text{mit} \qquad d_i, d_j \in D, \qquad d_i^{[x]} \leq d_j^{[x]} \tag{5.3}$$

Balance der Fläche Diese Formel berechnet die Balance ($B_{vertikal}, B_{horizontal}$) auf Basis der Summe der Abweichungen von den Mittelachsen des Dialoges (d_{ij}) (Nago et al. 2000). Die Balance gibt die Harmonie bzw. die Verteilung des optischen Gewichtes der Fläche an. Bei der Berechnung der Balance müssen verschiedene Eigenschaften berücksichtigt werden. Beispielsweise wirken dunkle Elemente (c= Farbe des Objektes) auf dem Bildschirm schwerer (Nago et al. 2000). Darüber hinaus macht es auch einen Unterschied wie groß (a= Größe des Objektes) die Objekte sind und welche Form (s= Form des Objektes) sie haben. Nago et al. (2000) beschreibt eine Methode, mit der die Berechnung der Balance durchgeführt werden kann. Als Basis dienen Farbe, Form und Größe des Objektes. Die Buchstaben L und R, bzw. T und B stehen für links und rechts, bzw. oben und unten.

$$B = 1 - \frac{|B_{vertikal}| + |B_{horizontal}|}{2} \tag{5.4}$$

$$B_{vertikal} = \frac{w_L - w_R}{max(|w_L|, |r_B|)}$$

$$B_{vertikal} = \frac{w_L - w_R}{max(|w_L|, |r_B|)}$$

$$B_{horizontal} = \frac{w_T - w_B}{max(|w_T|, |r_B|}$$

$$w_j = \sum_{i}^{n_j} d_{ij}(\frac{a_{ij}}{a_{max}} + |c_{ij} - c_{frame}| + s_{ij}); \quad j = L, R, T, B$$

In die Gewichtung der Abweichung fließen die Größe des Objektes und der Farbkontrast (Hintergrund und Objekt) mit ein. Nago et al. (2000) geben einen kontinuierlichen Wert zur Beschreibung der Farbe an. So liegt der Wert zwischen eins (Schwarz) und null (Weiß).

Angeordnetheit Das Maß gibt an, wie die Elemente auf dem Bildschirm zueinander ausgerichtet sind. Dies dient auch der Konsistenzberechnung. Je höher der Wert der Ausgewogenheit, desto ausgerichteter sind die Elemente zueinander (Mahajan und Shneiderman 1997). Für jedes Element werden die Kanten (oben und unten sowie links und rechts) analysiert, wobei Kanten mit gleichem Wert nur jeweils einmal gezählt werden (vgl. Abbildung 5.6). Das Ergebnis ist die Summe aller vorkommenden Kanten im Bildschirm (Mahajan und shneiderman 1997). Die Berechnung kann nur über eine Dialogfolge durchgeführt werden. Im Anschluss werden die beiden Dialoge betrachtet. Hamacher (2006) gibt folgende Formel zur Berechnung an (Dlg= Dialogfolge, L= Kanten links, R= Kanten rechts, T= Kanten oben, B= Kanten unten):

$$G(S) = \frac{|Dlg|}{|L| + |R| + |T| + |B|} \times 100 \tag{5.5}$$

$$\text{mit} \qquad L = l|l = d^{[x]}, \forall d \in D$$

$$R = r|r = d^{[x]} + d^{[w]}, \forall d \in D$$

$$T = t|t = d^{[y]}, \forall d \in D$$

$$B = b|b = d^{[y]} + d^{[h]}, \forall d \in D$$

Abbildung 5.6 zeigt die Anzahl der Kanten. Es sind acht vertikale und horizontale Kanten in dem Beispiel zu sehen.

Abbildung 5.6: Berechnung der Kanten

Dichte Die Dichte stellt ein Kriterium für die Menge der Elemente auf der Oberfläche dar. Nago et al. (2000) berechnen die Dichte (D) als Anteil der Fläche der Objekte (A_i) und

teilt diese durch die Gesamtfläche (A).

$$D = 1 - \frac{\sum A_i}{A} \tag{5.6}$$

Brechung der Oberflächenkonsistenz Brechung der Oberflächenkonsistenz: Ivory stellte 2001 zwei Maße zur Berechnung der Oberflächenkonsistenz (M) vor. Die Berechnung erfordert den Erwartungswert $E(M)$ und die Standardabweichung. Im folgenden Beispiel wird die Abweichung des Dialogs (m_i) von der Gesamtheit aller Dialoge einer Dialogfolge berechnet. Ivory gibt diesen Wert als Z-Score Maß an.

$$Z - \text{Score}(m_i) = \frac{m_i - E(M)}{\sigma_M} \qquad m_i \in M \tag{5.7}$$

Die Z-Scorewerte der einzelnen Dialoge werden miteinander verglichen. Nach Hamacher (2006) gilt in Anlehnung an Ivory (2001): *Die Normierung der Abweichung des Wertes vom Erwartungswert durch die Standardabweichung macht die Z-Scorewerte verschiedener Maße vergleichbar. Sind die Z-Scorewerte zweier verschiedener Maße gleich, so weichen sie im gleichen Verhältnis von ihrer Grundgesamtheit ab, dies geschieht unabhängig von Erwartungswert und eventueller Streuung des Maßes.* Wenn der Z-Scorewert innerhalb eines Intervalls von -1 und 1 liegt, dann wird der Wert als konsistent bezeichnet. Grund dafür ist, dass die Standardabweichung weniger vom Erwartungswert abweicht.

Zähler für Farbanzahl Bei diesem Verfahren wird die Anzahl der Farben (D_{col_F}), welche im Bildschirm vorkommen, gezählt, wobei die Hintergrundfarbe auch eine Rolle spielt. Bei Elementen, die sich überdecken, zählt das Vordere als Vordergrundfarbe, entsprechend das Hintere als Hintergrundfarbe. Die Empfehlung der maximalen Anzahl der Farben liegt zwischen 4 und 7 Farben (ISO 15008). Beim Einsatz mehrerer Farben wird vom System ein Signal gegeben, um den Benutzter entsprechend zu führen. Hamacher (2006) berechnet den Wert wie folgt:

$$N_{col_F} = |D_{col_F|} \tag{5.8}$$

$$\text{mit} \quad D_{col_F} = d_i^{[col]} | d_i \in D, \exists d_j \in D : d_i \text{ deckt } d_j \text{ ab}$$

Berechnung des Farbabstandes Der spektrale Abstand zweier Farben wird mit diesem Wert angegeben. Laut der Norm (ISO 15008) entspricht der RGB-Farbraum nicht dem menschlichen Sehempfinden, deshalb wurde der CIE YUV-Farbraum entwickelt (Hamacher 2006). Der Spektralabstand der Farben ist dem menschlichen Sehempfinden recht nahe.
Der E-Wert gibt die Distanz der Farben im CIE-Farbraum an. Ist der Farbabstand (E) größer als 100, so ergibt nach (Hamacher 2006) eine gut lesbare Farbkombination (Hamacher 2006). Die Farbgebung muss bei kleineren Werten neu angepasst werden. Der Benutzer muss jedoch prüfen, ob der geringe Abstand der Farbkombinationen aus Designgründen gewollt ist. Ferner ist auch die Rot-Blau-Kombination problematisch, diese sollte aus Usability-Sicht vermieden werden (DIN EN ISO 9241-8 Anforderungen an Farbdarstellungen), obwohl der Farbabstand mit E=144 deutlich über E=100 liegt.

Die Farben sind in der Rechnung mit i und j angeben.

$$\Delta E(Y_i u_i v_i, Y_j u_j v_j) = \sqrt{(155 * \frac{\Delta Y}{Y_{max}})^2 + (367 * \Delta u')^2 + (167 * \Delta v')^2} \tag{5.9}$$

$$\text{mit} \quad \Delta Y = |Y_i - Y - j|$$

$$Y_{max} = \max(Y_i, Y_J)$$

$$\Delta u' = |u_i - u_j|$$

$$\Delta v' = |v_i - v_j|$$

Die Abbildung 5.7 zeigt, dass links ein ausreichender spektraler Abstand und rechts ein zu kleiner spektraler Abstand gewählt wurde.

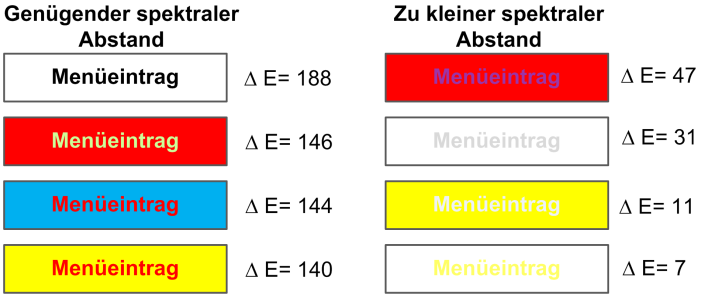

Abbildung 5.7: Spektraler Farbabstand (Hamacher 2006)

Nach Kingsbory (2003) können die Helligkeitsunterschiede zwischen zwei Farben mit folgender Formel berechnet werden: *Y=0,3*Rot+0,59*Grün+0.11*Blau.* Diese Werte entsprechen dem Helligkeitsempfinden des menschlichen Auges. Wobei die Farbwerte in einem Intervall zwischen *0* und *1* liegen können. Die Unterschiede zu Vorder- und Hintergrundfarbe werden ebenfalls berechnet.

$$D = |Yfg - Yhg| \tag{5.10}$$

Abstand der Farben ist mit D angegeben, Yfg= Vordergrundfarbe und Yhg= Hintergrundfarbe. Liegt der Abstand D unter der 0,3 Grenze, so ist es für das menschliche Auge schwierig, den Text auf der Hintergrundfarbe zu lesen.

Kontrastverhältnis Es wird der Helligkeitsunterschied zweier Farben im CIE-Farbraum betrachtet. Bei Menschen jüngeren Alters ist ein Kontrastverhältnis von 1 zu 2 anzustreben. Wohingegen bei älteren Menschen ein Kontrastverhältnis von 1 zu 4 gefordert wird. Bei Nacht im Fahrzeug liegt das geforderte Verhältnis zwischen 1.6 und 2 (ISO 15008). Die Vordergrundfarbe ist mit einem i und die Hintergrundfarbe mit einem j initialisiert.

Hamacher (2006) berechnet das Kontrastverhältnis wie folgt:

$$K(Y_i, Y_j) = \frac{\max(Y_i, Y_j)}{\min(Y_i, Y_j)} \quad \min(Y_i, Y_j) \neq 0 \tag{5.11}$$

Spektrale Distanzen Kombinationen mit stark unterschiedlichen Wellenlängen ermüden das Auge des Fahrers im Fahrzeug sehr schnell, denn diese können vom Menschen nicht gleichzeitig scharf gestellt werden (ISO 15008). Zusätzlich empfindet der Fahrer diese Farbe aufgrund des Effektes als unangenehm. Daher müssen die Farbkombinationen rot und blau aufeinander liegend vermieden werden. Zur Berechnung kann der Abstand der Wellenlängen herangezogen werden.

Schriftgröße Die Schriftgröße im Fahrzeug sollte 11 Punkte nicht unterschreiten, kleinere Schriftgrößen würden den Fahrer zu stark ablenken (ISO 15008). Zur Berechnung verwenden Ivory und Hearst (2001) die Anzahl der Pixel der Buchstabenhöhe.

Informationsdesign

Die Beschriftung von Menü- und Bedienelementen spielt eine sehr bedeutende Rolle im Bereich der Bediensystementwicklung. Deshalb muss bei der Systementwicklung die textuelle Kohärenz berücksichtigt werden. Im Folgenden werden Möglichkeiten der Formalisierung aufgezeigt. Es werden für diesen Bereich keine konkreten Formalisierungsregeln in der Literatur beschrieben. Dennoch sollen in diesem Abschnitt Möglichkeiten angedacht werden, um auch den Entwickler im Bereich des Informationsdesigns zu unterstützen.

Längenprüfung Um eine Längenprüfung durchführen zu können, müssen die Variablen *Feldlänge*, *Schriftart*, *Schriftgröße*, *Schriftschnitt* des Textes zur Verfügung stehen. Durch die unterschiedlichen *Laufweiten* sowie die unterschiedlichen *Schriftschnitte* (Fett, Kursiv) reicht es nicht aus, die einzelnen Buchstaben abzuzählen. Nichtproportionale oder dicktengleiche Schriftarten[4] bilden hier eine Ausnahme (Forssman und de Jong 2004). Um die pixelgenaue Länge eines Textes zu berechnen, müssen die Pixel der Zeichen (abhängig von Schriftart,-schnitt und -größe) aufaddiert werden.

Erkennung von Überschriften Die PCM-Bildschirme müssen mit einer Überschrift in der oberen Menüleiste versehen werden, um so eine eindeutige Kennung zu erhalten. Fehlt diese Überschrift, muss das System die Entwickler warnen.

Rechtschreibprüfung Für die Prüfung der Rechtschreibung kann auf eine herkömmliche Wörterbuchüberprüfung zurückgegriffen werden. Diese muss nur eine Schnittstelle anbieten, um von anderen Programmen wie beispielsweise MetaEdit+ eingelesen werden zu können.

[4] In nichtproportionalen oder dicktengleichen Schriftarten besitzt jedes Zeichen die gleiche Breite, sie wurden hauptsächlich als Schreibmaschinenschrift bekannt.

Prüfung der Begriffshierarchien Mithilfe einer Ontologie können beispielsweise Begriffshierarchien für die semantische Anordnung von Begrifflichkeiten bei der Gestaltung von Fahrerinformationssystemen erstellt werden. So kann über einen Oberbegriff der Unterbegriff semantisch zugeordnet werden (Hamacher 2006). Das Informationsdesign kann mittels Ontologien geprüft werden, hierbei ist denkbar, dass ein Navigationsbaum mit allen Funktionen in einem Textmanagementprogramm abgelegt wird. Die Entwickler erhalten so Unterstützung bei der Gestaltung von Bedienkonzepten.

Interaktionsdesign

Die Kontrolle der Interaktion wird meist mit sogenannten *Statecharts* durchgeführt. Wobei es derzeit sehr wenige Werkzeuge gibt, die eine Überprüfung des *Interaktionsdesigns* ermöglichen. Language Workbenches, wie sie in der Werkzeugkette zum Einsatz kommen, bieten dafür einige Ansatzpunkte (Heilig 2007). Als Grundlage für die Berechnung des *Interaktionsdesigns* durch die entsprechenden Maße dienen Transitionen, wie beispielsweise die Verbindung zwischen zwei PCM-Bildschirmen. Für die Berechnung des *Interaktionsdesigns* können folgende Formeln herangezogen werden.

Streuung von Dialogfolge Der *Coefficient of Variation* $CoV(M)$ berechnet die Variation der Werte eines Maßes über die vollständige Dialogfolge. Problematisch ist, dass dieser Wert ein reines Vergleichsmaß ist, daher müssen Referenzwerte von zwei vollständigen Dialogfolgen vorliegen. Diese müssten im System als obere bzw. untere Schranke abgespeichert werden. Somit könnte die aktuell gestaltete Dialogfolge mit den Referenzwerten verglichen werden. Je kleiner die Streuung der Werte (Erwartungswert), desto kleiner ist der Koeffizient. Dies hat zur Folge: Je kleiner der Koeffizient desto konsistenter ist der Dialog (Ivory 2001).

$$CoV(M) = 100 * \frac{\sigma_m}{E(M)} \tag{5.12}$$

Nach Hamacher (2006) eigen sich zum Beispiel für die Berechnung des *Coefficient of Variation* folgende Maße: *Display Aspect Ratio*, *Anzahl der Displayelemente*, *Anzahl der Farben*, *Angeordnetheit*, *Balance* und der A*bstand der Randpunkte*.

Menütiefe Zapharis et al. (2002) geben eine optimale Menütiefe von 7 Ebenen an. Zur Berechnung werden die Menüeinträge N_{me_i}, welche an einem Baum zusammen hängen, aufaddiert me_i (Hamacher 2006).

$$N_{me_i} = |me_i| \tag{5.13}$$

Erkenntnisse für diese Arbeit

Ziel dieser Analyse war herauszufinden, ob es existierende Formeln und Metriken zur Operationalisierung der Bediensystemgestaltung gibt. Denn für die automatische Prüfung von Benutzerschnittstellen ist es notwendig, die Elemente der Benutzeroberfläche zu formalisieren. Die meisten Metriken für den Bereich des *Oberflächendesigns* lassen in dem das Konzept der Werkzeugkette abbilden. Die Metriken können in einer Art Regelbasis abgespeichert und vom GUI-Builder mittels eines Berechnungsframeworkes überprüft werden. Wobei die Berechnung der Dichte nicht *on-the-fly* durchgeführt werden kann, denn hierzu müssen alle Elemente platziert sein. Im Bereich des *Informationsdesign* finden sich sehr wenige Metriken in der Literatur. Hier können mittels Ontologie die Hierarchien der Begrifflichkeiten festgestellt werden. Ferner kann auf Software zurückgegriffen werden, die eine Überprüfung der Rechtschreibung durchführt. Anders erscheint es bei den Metriken für das *Interaktionsdesign*. Nach der Untersuchung des CASE-Tools MetaEdit+ können zwar die Formeln nicht direkt abgebildet werden, aber das Werkzeug bietet über API und eine Skriptsprache die Möglichkeit, die Menütiefe zu errechnen.

5.4.2 Integration des Konzepts in die Werkzeugkette

In den vorausgegangenen Abschnitten wurden verschiedene Methoden vorgestellt, die den Entwickler bei interaktiven Bediensystemen unterstützen. Ausgehend von diesen Erkenntnissen wird das Konzept für die Unterstützungskomponente vorgestellt.
In der Werkzeugkette wird die Dimension *Oberflächendesign* mit einem GUI-Builder namens *JFormDesinger* realisiert (Heilig 2007). Die Dimensionen *Informations-* und *Interaktionsdesign*, bei denen die modellgetriebene Softwareentwicklung zum Einsatz kommt, werden mithilfe des CASE-Tools MetaEdit+ abgebildet (vgl. Abschnitt 5.3). Eine ausführliche Beschreibung des CASE-Tools wird in den Arbeiten von Bock (2007) und Heilig (2007) vorgenommen. In Rahmen dieser Arbeit wird jedoch nicht in der Dimension *Oberflächendesign* auf den GUI-Builder zurückgegriffen, wie in den Arbeiten von Bock (2007) und Heilig (2007), sondern auf das Werkzeug *Microsoft Expression Blend*, da dieses umfangreiche Möglichkeiten bietet, komplexere Grafiken zu erstellen. Durch die ständige Entwicklung neuer Funktionen im Fahrzeug wächst auch die Komplexität der Grafiken. Darüber hinaus hat der *Expression Blend* den Vorteil des vertrauten Erscheinungsbildes, da dieser sehr einem Bildbearbeitungsprogramm ähnelt, wodurch die Akzeptanz bei den *Designer* gefördert wird. Der Hauptgrund für die Verwendung des *Expression Blend* in dieser Arbeit ist jedoch, dass er das XAML-Format (e**X**tensible **A**pplication **M**arkup **L**anguage) als Beschreibungssprache für grafische Oberflächen verwendet. Dadurch ist die erstellte Oberfläche jederzeit als XAML- bzw. XML-Dialekt maschinenlesbar vorhanden und die enthaltenen Informationen, die für die Berechnung der Metriken notwendig sind, werden somit verfügbar (vgl. Abschnitt 5.4.2). Mithilfe von XAML kann der *Designer* Grafiken vektorbasiert selbst erstellen. Dadurch sind Oberflächen mit High-End-Content wie 3D-Element-Rendering, Schattenwurf, Spiegelungen, und vektorbasierte Zeichnungen möglich, die den *Designer* unterstützen. Der *Microsoft Expression Blend* bietet eine übersichtliche Trennung zwischen dem Quelltext und der Oberfläche. Die erstellten Grafiken werden dann vom CASE-Tool mit dem *Interaktions-*

und *Informationsdesign* für die Erstellung von Prototypen versehen. Somit ist der *Designer* enger in den Entwicklungsprozess miteinbezogen, wodurch der Prozess beschleunigt wird.

Erkenntnisse aus der Formalisierbarkeit der Dimensionen

Das *Oberflächendesign*, welches mit dem XAML-Format beschrieben wird, enthält Informationen, die für die Formalisierung der Oberfläche herangezogen werden können (vgl. Abbildung 5.8).

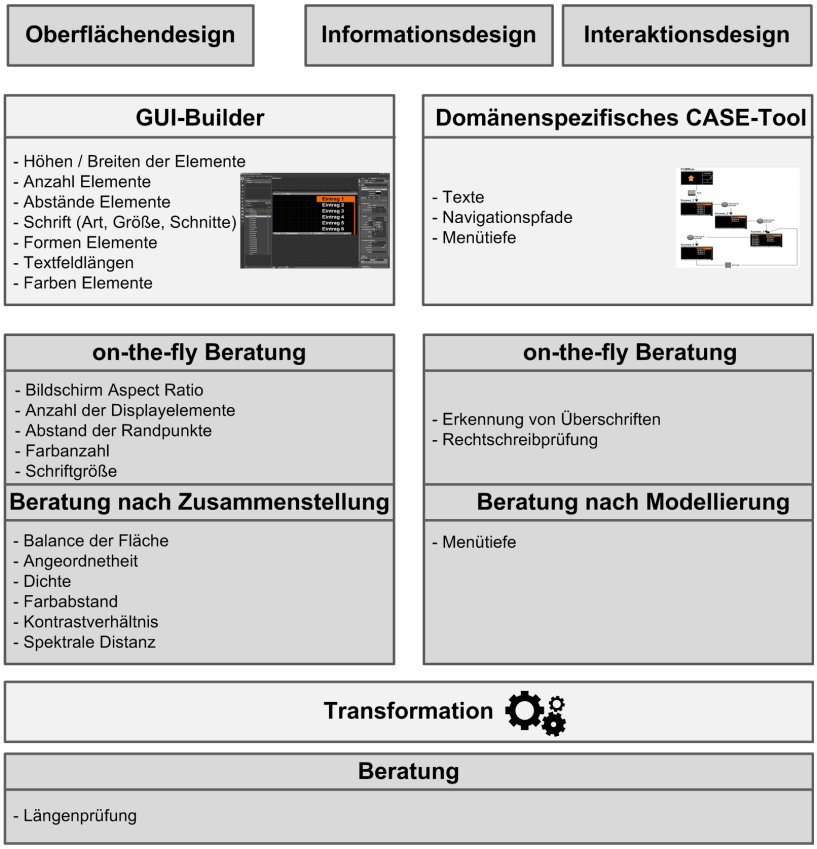

Abbildung 5.8: Klassifizierung der Beratung

So können aus dem XAML-Format die Variablen für: *Höhe* und *Breite* der Elemente, *Anzahl* der Elemente, *Abstände* der Elemente, *Schrift* (Art, Größe, Schnitte, Farbe), *Formen*, *Textlängen* und *Farben* der Elemente herausgelesen werden. Die Prüfung von Bildschirm *Aspect Ratio*, *Anzahl der Displayelemente*, *Abstand der Randpunkte*, *Farbanzahl* und *Schriftgröße* kann *on-the-fly* durchgeführt werden. Die restlichen Kriterien wie *Balance der Fläche*, *Angeordnetheit*, *Dichte*, *Farbabstand*, *Kontrastverhältnis* und *Spektrale Distanz* können erst nach der Bildung von sogenannten *atomaren Gruppen* (vgl. Abschnitt 5.4.1) geprüft werden. Das *Informations-* und *Interaktionsdesign* wird zum größten Teil mit dem domänenspezifischen CASE-Tool abgebildet. Zur Überprüfung dieser Dimensionen können die Variablen *Text*, *Navigationspfade* und *Menütiefe* herangezogen werden. Die Menütiefe kann erst am Ende einer Modellierung geprüft werden, da alle Objekte berücksichtigt werden müssen. Abbildung 5.8 zeigt die Variablen für das *Oberflächen-*, *Informations-* und *Interaktionsdesign* sowie die abgeleiteten Möglichkeiten der Beratung während der Entwicklung. Die Komponente *Informationsdesign* wurde eher dem CASE-Tool zugeordnet, da dort die Eingabe der

Beschriftungen erfolgt (vgl. Abbildung 5.8). Eine Ausnahme bildet die *Längenprüfung*. Diese kann die Entwickler nicht *on-the-fly* während der Gestaltung der Bediensysteme unterstützen, da die *Art* und *Größe* der Schrift im GUI-Builder definiert wird und die *Länge* der Texte (z. B. Menüeinträge) im domänenspezifischen CASE-Tool (vgl. Abbildung 5.8). Um die Längenprüfung dennoch anbieten zu können, muss eine Transformation zwischen beiden Anwendungen durchgeführt werden, die zum einen die Schrift mit ihren Eigenschaften, wie *Art*, *Größe*, *Schnitte* und zum anderen die Länge der Texte abbildet (vgl. Abbildung 5.8).

Oberflächendesign

Im Bereich des *Oberflächendesigns* ist die Anordnung der Elemente eines der wichtigsten Designkriterien. Das *Oberflächendesign* wird, wie im letzten Abschnitt beschrieben, mit dem XAML GUI-Builder erstellt. Im Folgenden wird nun ein Konzept vorgestellt, welches die *Designer* schon während der Gestaltung von Oberflächen im HMI-Entwicklungsprozess unterstützt, indem beispielsweise ergonomisch gestaltete Bildschirme in einer Art Bibliothek vorliegen. Die parametrisierbaren Designregeln wurden von Abschnitt 5.4.2 übernommen. Die gezeigten Mockups dienen zur Verdeutlichung des Konzepts, da die Integration per Plug-ins derzeit noch nicht möglich ist. Die Methoden zur Design-Unterstützung werden in sogenannten Panels dargestellt, die frei auf der Benutzeroberfläche, ähnlich wie bei Photoshop, bewegt werden können. Diese werden in *Appearance Panel*, *Geometry-Control Panel*, *Color-Control Panel* und *Font-Control Panel* unterteilt. In den einzelnen Panels können bei Bedarf Hilfetexte zu den Kriterien angezeigt werden. Die Panels werden anhand von Mockups, welche grafisch aufbereitete sind, näher erläutert, um das Konzept zu verdeutlichen. Die *Designer* haben nach dem Start des Programmes die Möglichkeit, die zuvor definierten Projekte aus einer Art Bibliothek aufzurufen. D. h., wählt ein *Designer* zum Beispiel das Projekt PCM 2.1, so werden alle Panels mit den zuvor definierten Gestaltungskriterien eigens für dieses Projekt geladen und nur die Farben im Panel angezeigt, welche mit dem Produkt-Styleguide konform sind. In Anhang C wird eine PCM-Bildschirm zeigt, welcher mit dem XAML GUI-Builder implementiert wurde.

Geometry-Control Panel Das wichtigste Merkmal bei der Gestaltung von interaktiven Bediensystemen ist die Konsistenz der Oberfläche (Johnson 2000). Eine Methode dabei ist, die Position der Elemente mithilfe von Rastern oder Hilfslinien einheitlich zu gestalten. Diese Form von Konsistenzlinien wird *Snap-Grid* genannt. Mithilfe eines Dialoges kann der *Designer* die Elemente entsprechend ausrichten. Diese Funktion ist bei der Gestaltung von Menüeinträgen sehr sinnvoll. Die Funktion unterstützt auch die Ausrichtung von Textelementen. Werden die Elemente ausgewählt und verschoben, so wird immer der aktuelle Abstand zu den anderen Elementen angezeigt. Zusätzlich wird der Abstand zum Rand angezeigt (vgl. Geometry Panel Margin), denn nach der ISO 15008 muss eine bestimmte Anzahl von Pixeln zum Rand frei gehalten werden. Ferner werden den Entwicklern im *Geometry-Control Panel* der *Aspect Ratio* (Verhältnis zwischen Höhe und Breite) und die Anzahl der Bildschirmelemente aufgezeigt. Die *Dichte* und die *Balance der Fläche* kann erst zum Schluss geprüft werden, da dafür die Anzahl aller Elemente, die sich auf dem Bildschirm befinden, benötigt wird. Gleiches gilt auch für den *Gridedness-Wert* (Angeordnetheit). Die Formel

prüft, wie viele unterschiedliche Kanten im Objekt vorkommen (vgl. Abschnitt 5.4.2). Diese Werte können wie bei der IDA-Qualitätssicherungskomponente (vgl. Abschnitt 4.3.1) im Anschluss an die Gestaltung per Knopfdruck ausgeführt werden. Folgende Abbildung 5.9 zeigt die XAML-GUI-Builder mit dem integrierten Geometry-Control Panel (Mockup).

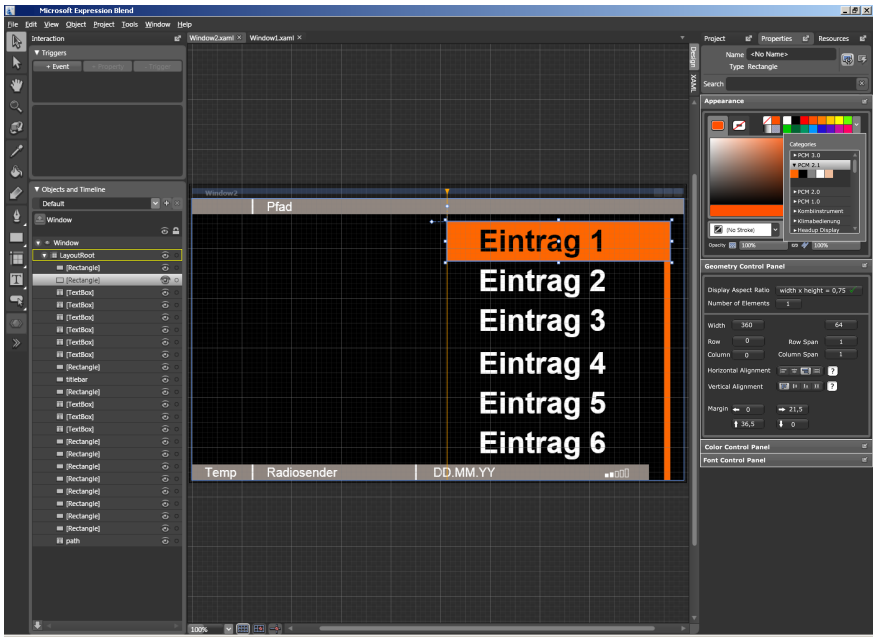

Abbildung 5.9: Mockup-XAML GUI-Builder mit Geometry-Control und Appearance Panel

Appearance Panel Das Panel zeigt die verschiedenen Farbpakete an, sie sind gruppiert nach Projekten. Abbildung 5.9 zeigt die Farben an, die nach dem Styleguide der Porsche AG empfohlen werden. Natürlich kann der *Designer* auch andere Farben auswählen. Er soll in seiner Kreativität nicht eingeschränkt werden.

Color-Control Panel Mithilfe dieses Panels wird der *Designer* im Bereich der Farbgebung unterstützt. Nach der Auswahl einer Farbe zeigt das Panel an, für welche Bereiche diese Farbe verwendet werden darf. Abbildung 5.10 zeigt, dass die Farbe *Weis* für Menüeinträge und für Überschriften verwendet werden darf. Bei der Prüfung der Farbgebung müssen wie in Abschnitt 5.4.1 beschrieben *atomare Gruppen* gebildet werden.

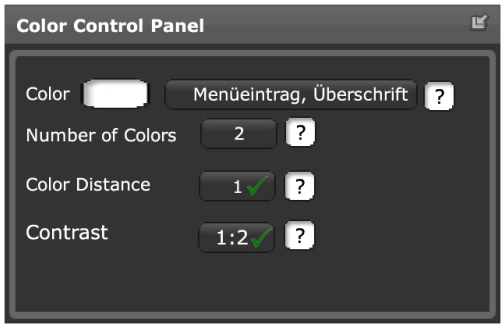

Abbildung 5.10: Mockup Color-Control Panel

Am folgenden Beispiel soll gezeigt werden, wie der Farbabstand zwischen zwei Farben be-

rechnet wird. Dabei wird die in Abschnitt 5.4.1 beschriebene Vorgehensweise von Beirekdar et al. (2002) angewendet. Zuerst werden die Zielelemente ausgewählt. Im nächsten Schritt werden die Elemente, die es zu prüfen gilt, zusammengestellt. Die zusammengestellten Elemente bilden eine atomare Gruppe. Im letzten Schritt werden die Prüfungsregeln auf die atomare Gruppe angewendet. Es soll geprüft werden, ob der Farbabstand zwischen der Hintergrundfarbe des PCM-Bildschirmes und einem Menüeintrag in Ordnung ist. Zur Berechnung wird die Formel *Y=0,3*Rot+0,59*Grün+0.11*Blau* von Kingsbory (2003) verwendet. Im Folgenden werden nun die Schritte erläutert, die durchgeführt werden müssen.
Zu Beginn werden die Zielelemente *Eintrag1* und der *Hintergrund* vom *Designer* ausgewählt. Im nächsten Schritt werden die beiden Elemente (*Eintrag1* und *Hintergrund*)eingelesen. Um hierbei in der Struktur gezielt nach einem Element suchen zu können, wird das Document Object Model (DOM) verwendet. Somit muss der Quelltext nicht sequenziell von vorne nach hinten durchlaufen werden. Die beiden Elemente bilden jetzt eine atomare Gruppe. Im XAML-Quelltext sehen die Elemente wie folgt aus:

```
<Rectangle Fill="#FF000000" Stroke="FF000000" RadiusX="0" RadiusY="0"
    Margin="0,0,56,0" VerticalAlignment="Bottom" Height="25"/>
```

Listing 5.1: Quelltext-XAML Hintergrund

```
<TextBox HorizontalAlignment="Right" Margin="0,39.5,0,0"
    VerticalAlignment="Top" Width="333.5" Height="64" Background="{x:Null
    }" BorderThickness="0,0,0,0" FontFamily="Arial" FontSize="12"
    FontWeight="Bold" Foreground="#FFFFFFFF" Text="Eintrag 1&#xd;&#xa;"
    TextWrapping="Wrap"/>
```

Listing 5.2: Quelltext-XAML Eintrag 1

Im letzten Schritt werden Mithilfe des Berechnungsframeworks die Metriken, welche in der Regelbasis abgelegt wurden, berechnet. Zu Beginn werden die Werte von hexadezimal in dezimal Werte umgerechnet. Die ersten beiden Ziffern (hexadezimal) geben den Alphawert an, dieser wird nicht berücksichtigt, da es sich um keine transparente Fläche handelt. Der Hintergrund ist Schwarz (R=0, G=0, B=0) und der Eintrag 1 ist Weiß eingefärbt (R=1, G=1; B=1). Im Anschluss folgt die Berechnung des Farbabstandes (Hintergrund Y_b, Menüeintrag Y_f).
$Y_b = 0,3 * 0 + 0,59 * 0 + 0,11 * 0 = 0$
$Y_f = 0,3 * 1 + 0,59 * 1 + 0,11 * 1 = 1$
$D = |Y_b - Y_f| = 1$
Der berechnete Abstand der Farben liegt bei *1.* Das bedeutet, er ist für das menschliche Auge sehr gut sichtbar. Somit erscheint neben dem errechneten Wert ein grüner Haken (vgl. Abbildung 5.10). Die Berechnung für das Kontrastverhältnis und die spektrale Distanz ist anlog zu der Vorgestellten.

Der folgenden Quelltext zeigt die imlpementierte Berechnung des Farbabstandes. Nach dem Auslesen der Elemente wird die Umrechnung von hexadezimal in dezimal vorgenommen. Auf das Ergbnis wendet das Berechnungsframework die Metrik an, welche von der Regelbasis bereitgestellt wurde.

```
string backColor = "";
string frontColor = "";

// XAML Datei zum Lesen öffnen
XmlTextReader xmlTextReader = new XmlTextReader("screen.xaml");
while(xmlTextReader.Read())
{
    // Alle Elemente abfragen
    if (xmlTextReader.NodeType == XmlNodeType.Element)
    {
        // Farbe des Hintergrundelementes auslesen
        if (xmlTextReader.Name == "Rectangle")
            backColor = xmlTextReader.GetAttribute("Fill");

        // Farbe des Textes im Vordergrund auslesen
        if (xmlTextReader.Name == "TextBox")
            frontColor = xmlTextReader.GetAttribute("Foreground");
    }
}
if (backColor != "" && frontColor != "")
{
    float backR = Convert.ToInt32(backColor.Substring(3,2),16)/255f;
    float backG = Convert.ToInt32(backColor.Substring(5,2),16)/255f;
    float backB = Convert.ToInt32(backColor.Substring(7,2),16)/255f;
    float frontR = Convert.ToInt32(frontColor.Substring(3,2),16)/255f;
    float frontG = Convert.ToInt32(frontColor.Substring(5,2),16)/255f;
    float frontB = Convert.ToInt32(frontColor.Substring(7,2),16)/255f;

    // Abstandsberechnung
    float Y1 = 0.3f * backR + 0.59f * backG + 0.11f * backB;
    float Y2 = 0.3f * frontR + 0.59f * frontG + 0.11f * frontB;
    float DeltaY = Y2 - Y1;

    // Ausgabe
    System.Console.WriteLine("Farbabstand {0} zu {1} ist {2}", backColor,
        frontColor,DeltaY);
    System.Console.ReadLine();
}
```

Listing 5.3: Berechnung des Farbabstandes

Anhand der Berechnung des Farbabstandes wurde das Konzept, welches im Rahmen der Arbeit vorgestellt wurde, auf seine Realisierbarkeit überprüft. Da dies zu einem positiven Ergebnis führte, wird davon ausgegangen, dass die Berechnung auf andere Dimensionen des Oberflächendesigns übertragbar ist.

Font-Control Panel Dieses Panel unterstützt den *Designer* bei der Erstellung von neuen Menüeinträgen. So zeigt das Panel die aktuell gewählte Schriftart sowie die Schriftgröße an (vgl. Abbildung 5.11). Darüber hinaus wird die Anzahl der verwendeten Schriften errechnet. Wurde vom *Designer* die richtige Schrift sowie die richtige Größe ausgewählt, so bekommt er durch das Font-Control Panel Rückmeldung in Form eines grünen OK-Häkchen. Für diese Berechnung gibt es Programmbibliotheken[5]. Diese Länge kann dann mit der Feldlänge verglichen werden.

Abbildung 5.11: Mockup Font-Control Panel

Bibliothek Über eine Bibliothek kann der *Designer* die zuvor abgelegten ergonomischen Widgets, wie etwa PCM 2.1 Bildschirme, per *Drag and Drop* an der gewünschten Stelle platzieren (vgl. Abbildung 5.12). Diese Funktionen bietet der GUI-Builder derzeit schon an. Die Funktion ist vergleichbar mit den Shapes, welche in Microsoft Visio erstellt werden können. Darüber hinaus wurde diese Funktion explizit von den Entwicklern bei der Befragung verlangt (vgl. Kapitel 2). Die Bibliothek bietet neben der gruppierten Darstellung der Elemente eine Suchfunktion mit Texteingabe.

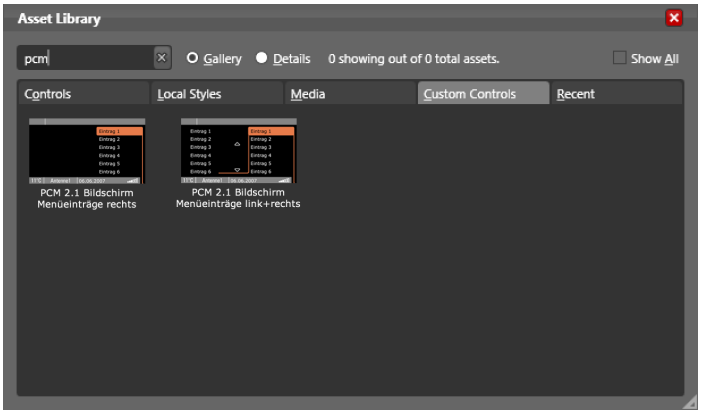

Abbildung 5.12: Bibliothek Projekt PCM 2.1

Informationsdesign

In diesem Abschnitt wird erläutert, wie die Entwickler im Bereich des *Informationsdesigns* unterstützt werden können. Hierbei kommt das Werkzeug MetaEdit+ zum Einsatz.
Will der Entwickler einen neuen Bildschirm in die aktuellen Bedienabläufe integrieren, so wird der Bildschirm einfach per *Drag and Drop* an die gewünschte Stelle gezogen. Sobald der Bildschirm korrekt positioniert wurde, erscheint ein Eingabefeld, in dem der Name, die Kopfzeile und die Menüeinträge eingegeben werden können. Hierbei sind der Name und die

[5]In JAVA gibt es die Klasse *FontMetrics*, welche die Textlänge mit der Feldlänge vergleicht.

Kopfzeile des Bildschirms Pflichtfelder. Werden diese nicht ausgefüllt, erscheint ein Warnhinweis in Form eines Pop-Ups. Über die Kopfzeile lassen sich zudem die Navigationspfade der Menüstruktur erkennen. Für die Eingabe der Beschriftungen, wie beispielsweise der Menüeinträge, können dem Entwickler weitere Unterstützungsfunktionen im CASE-Tool angeboten werden.
Im Bereich des *Informationsdesigns* kann die Rechtschreibprüfung in Form einer Bibliothek über die API in das MetaCASE-Tool integriert werden. Einige Hersteller bieten solche Korrekturhilfen in Form von Bibliotheken bereits an, um so die Fehlerquellen in der Rechtschreibung zu vermeiden. Die Anwendung JRSpell[6], welche über die API von MetaEdit+ integriert werden kann, prüft beispielsweise Textstrings und Textboxen auf korrekte Rechtschreibung. Des Weiteren können problemlos Wörterbücher in verschiedenen Sprachen heruntergeladen werden. Dieser Aspekt ist für die Porsche AG von besonderer Bedeutung, da der Konzern viel für das Ausland produziert. Ferner könnte über die API von MetaEdit+ eine Wortergänzung umgesetzt werden. Diese ist in der Arbeit nicht implementiert, sondern soll nur an dieser Stelle angeregt werde, um darzustellen, wie Entwickler zukünftig unterstützt werden könnten. Ziel der Entwickler ist es beispielsweise, ein neues Menükonzept für den Hauptbedienbereich Navigation zu erstellen. Bei der Eingabe der Beschriftung von Menüeinträgen wird der Buchstabe *Z* eingegeben. Die Anwendung sucht nun in der Datenbank bzw. im Textverwaltungsprogramm nach Wörtern, die mit demselben Muster beginnen, und zeigt diese in einem Popup-Menü an. Wenn der eingegebene Text vorkommt, könnte gleich neben dem Dialog die Übersetzung des gefundenen Musters stehen. Diese Funktion spart bei der Entwicklung enorm Zeit und vermeidet die Verwendung von Synonymen.
Abbildung 5.13 zeigt eine mögliche Darstellung der angedachten Wortergänzungsfunktion.

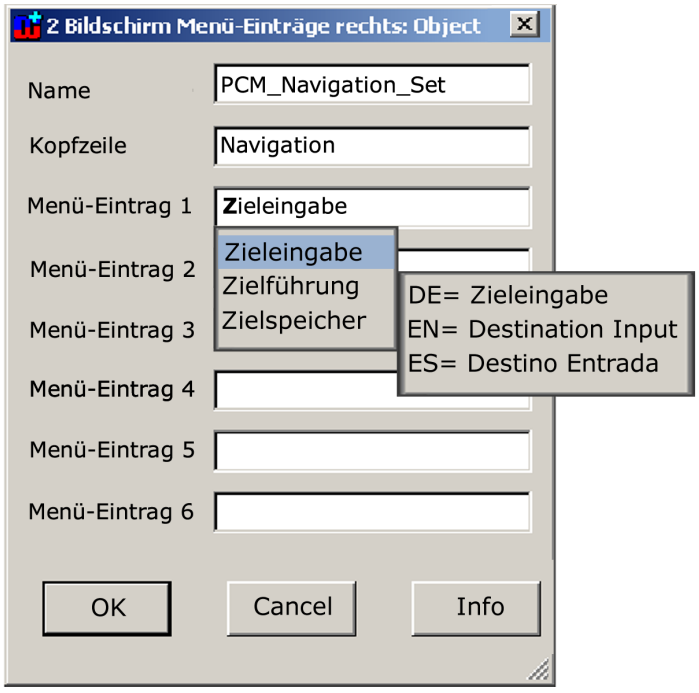

Abbildung 5.13: Mockup-Wortergänzung

[6]Die Methode wird SpellCheckWords genannt (http://www.jrsoft.de/ Letzter Zugriff: 03.08.2007).

Die *Längenprüfung* der Beschriftungen muss mittels einer Transformation der Metriken aus dem GUI-Builder und dem domänenspezifischen CASE-Tool durchgeführt werden. Um die Länge der Beschriftungen zu prüfen, müssen die Variablen *Feldlänge*, *Schriftart*, *Schriftgröße*, *Schriftschnitt* und der *Text* bekannt sein (vgl. Abschnitt 5.4.1). Die zuletzt genannten Parameter müssen in MetaEdit+ eingelesen werden. Alle zuvor genannten Variablen bis auf die Variable Text werden im GUI-Builder definiert. Die Eingabe der Texte, wie beispielsweise die Menüeinträge, erfolgen im CASE-Tool. Aus diesem Grund findet die Längenprüfung erst bei der Eingabe des Textes im CASE-Tool statt. Derzeit ist es nicht möglich, die Längenprüfung, wie oben beschrieben, durchzuführen, da MetaEdit+ nicht die benötigte Schnittstelle zu Verfügung stellt. Abbildung 5.14 soll dennoch das angedachte Konzept der Längenprüfung verdeutlichen. Die Beschriftung Navigation Werkseinstellung ist zu lang für das Objekt von Typ *Bildschirm Menü-Einträge rechts*.

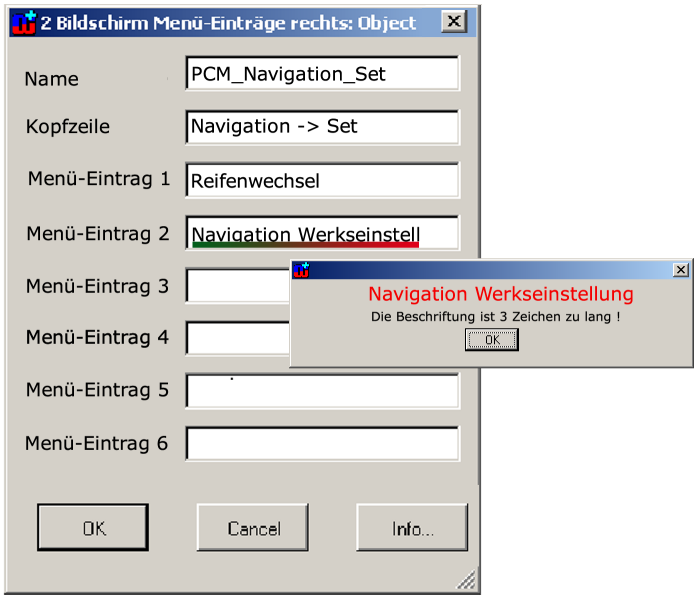

Abbildung 5.14: Mockup Längenprüfung der Beschriftung

Alternativ kann derzeit der Graph (vgl. nächster Abschnitt)als XML-Format exportiert werden, so dass im Anschluss ein Parser die Menüeinträge (TAGs) auf ihre zuvor definierte Länge prüft. Hierbei kann aber nur die *Stringlänge* definiert werden. Außerdem werden die Variablen (*Feldlänge*, *Schriftart*, *Schriftgröße*, *Schriftschnitt*) aus dem GUI-Builder nicht berücksichtigt.

Interaktionsdesign

Das Interaktionskonzept wird von den *Usabiliy-Engineers* mithilfe des domänenspezifischen CASE-Tools oder auch Language Workbenche (MetaEdit+) umgesetzt. Bevor die Unterstützungsmöglichkeiten aufgezeigt werden können, müssen zunächst einige Begrifflichkeiten geklärt werden, um so den Aufbau und die damit verbundenen Möglichkeiten für die Unterstützung des *Interaktionsdesigns* verstehen zu können. Das Meta-Modellierungswerkzeug verwendet für die Abbildung ein Metamodell namens *GOPPRR* (Bock 2007). *GOPPRR* steht für *Graphs*, *Objects*, *Properties*, *Port*, *Realitionsships* und *Roles*. Im Folgenden werden die einzelnen Komponenten kurz erläutert.

Graph: Der Graph beschreibt das Metamodell einer domänenspezifischen Sprache. So beinhaltet beispielsweise ein Graph wie die Bildschirme (Objekte) miteinander verbunden (Bindings) sind.

Objects: Objekte sind zentrale Elemente mit dessen Hilfe ein Domänenkonzept abgebildet werden kann (Bock 2007). Den Objekten können mehrere Eigenschaften (Properties) zugeordnet werden. Das Objekt *PCM-Bildschirme* besitzt die Properties: *Bildschirm MAIN*, *Bildschirm Menüeintrag rechts* und *Bildschirm Untermenüeintrag links*.

Properties: Jedem Element im Graph kann eine Eigenschaft gegeben werden, wie beispielsweise dem Objekt Bildschirm der Datentype *String* für die Menüeinträge.

Ports: Die Objekte können mit sogenannten Ports versehen werden. Die Anschlüsse definieren den Ein- und Ausgang eines Objektes.

Relationships: Die Beziehung formalisiert die Verbindung der Objekte untereinander. D.h. die Bildschirme *A* und *B* sind über die *Hauptbedientaste Tel* miteinander verbunden.

Roles: Roles legen fest, welche Rolle ein Objekt in einer Beziehung einnimmt. D.h. mit der *Hauptbedientaste Tel* (Relationship) gelangt der Benutzter beispielsweise zum Eingang von *Bildschirm B*.

Die folgenden Abbildung 5.15 zeigt die möglichen Transitionen.

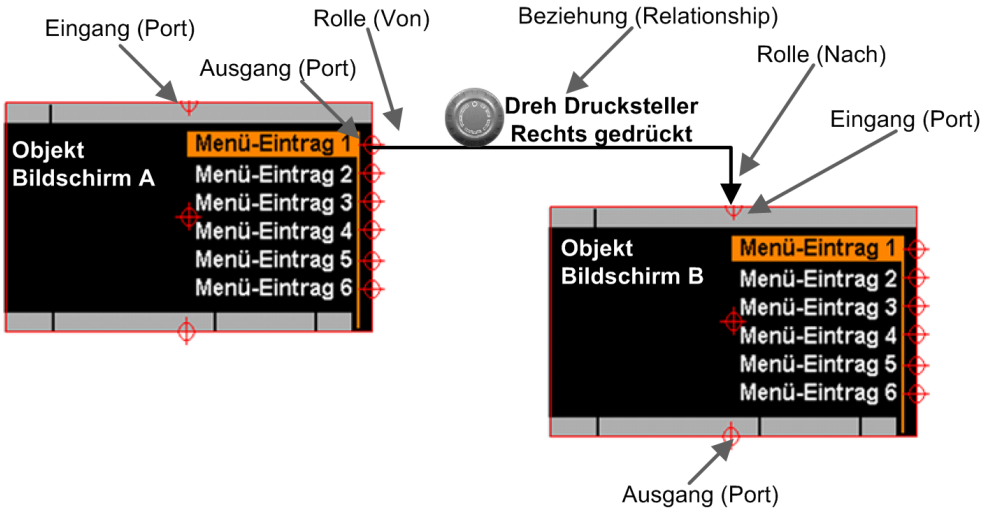

Abbildung 5.15: Transitionen in MetaEdit+

Der *Usability-Engineer* und der *technische Experte* können in enger Abstimmung die erlaubten Verbindungen definieren. D.h. die Zustandsübergänge werden eingeschränkt, um so ein

einheitliches *Interaktionsdesign* gewährleisten zu können. Hierbei könnte der *Programmierer* nach Absprache mit dem *Usability-Engineer* und *technischen Experten* die Einschränkungen im Modell vornehmen. Diese Einschränkung unterstützt die Bediensystementwickler insofern, dass sie nur mit bestimmen Bedienknöpfen (z. B. Hauptbedientaste) an den Eingang (Port) eines anderen Bildschirms gelangen können.
Das System lässt nur die zuvor spezifizierten Verbindungen zu und erzeugt im Bedarfsfall Hinweisfenster mit entsprechenden Informationen (vgl. Abbildung 5.16). So können die Entwickler aus dem Hilfetext entnehmen, was sie falsch gemacht haben.

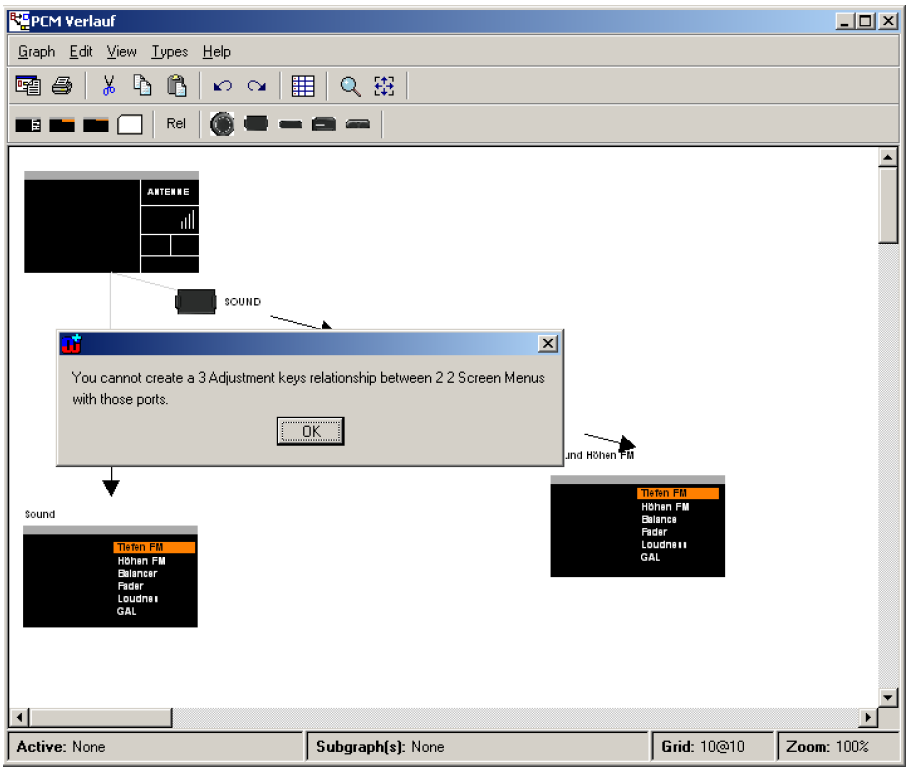

Abbildung 5.16: Fehlermeldung bei Benutzung von falschen Bedienelementen

Somit ist genau definiert, mit welchen Bedienelementen die Eingangs- und Ausgangsports belegt werden dürfen, um die PCM-Bildschirme miteinander zu verbinden. Es wird genau festgelegt, auf welche Art und Weise die Bildschirme miteinander verbunden werden dürfen. Ferner lassen sich über sogenannte *Connectivity Constraints* Verzweigungen zu genau einem Zielbildschirm spezifizieren. Außerdem können Domänenregeln so spezifiziert werden, dass nur spezielle Interaktionselemente, wie beispielsweise, die *Zifferntasten* auf bestimmten Menübildschirmen angeboten werden, wie der Bildschirm mit der Telefonfunktion. In diesem Fall dient der *Ziffernblock* als *Wähltaste*. Des Weiteren können Eigenschaften für die Relationsships definiert werden. Das folgende Beispiel zeigt, dass die Hauptbedientaste nur mit den Funktionen *MAIN*, *AUDIO*, *SOUND*, *TEL*, *MAIL*, *INFO*, *TRIP* und *NAVI* belegt werden kann. D.h., den Bediensystementwicklern werden über ein *Drop-Down* Menü die erlaubten Verbindungen zur Auswahl angeboten (vgl. Abbildung 5.17). Darüber hinaus wird die zulässige Transition mit Reitern dargestellt. D.h. der *technische Experte* darf nur mit einer Hauptbedientaste vom aktuellen *Bildschirm-PCM Main* zum *Bildschirm 1* eine Relationship durchführen.

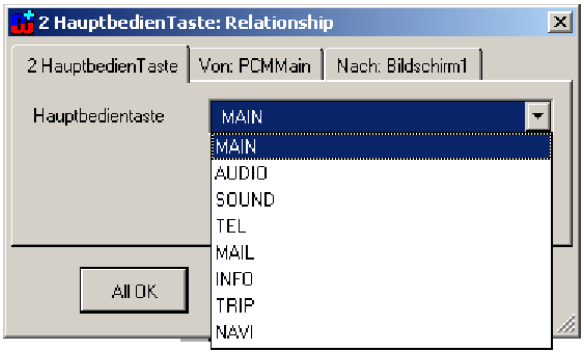

Abbildung 5.17: Erlaubte Transition der Hauptbedientaste

Die neuste Version MetaEdit 4.5+ ermöglicht es die Menütiefe mit der Verwendung von Rollen und Reports zu ermitteln. MetaEdit+ kann die Verknüpfungen (Verbindung zwischen zwei Bildschirmen) schon während der Modellierungszeit prüfen. Das bedeutet, über die Reports können die Vorgängerbildschirme, welche sich durch die Rollen genau spezifizieren lassen, zur Laufzeit abgezählt werden. Das bedeutet, dass die Anzahl der Vorgängerbildschirme genau ermittelt werden kann. Wählt beispielsweise der *technische Experte* eine neue Verbindung zu seinem aktuellen Bildschirm, so kann von MetaEdit+ geprüft werden, wie viele Vorgänger der Bildschirm hat. Es erscheint ein Warnhinweis, wenn zu viele Vorgänger Bildschirme (>=7) im aktuellen Menü sind, d.h. ist die maximale Menütiefe von 7 erreicht, so ist es nicht mehr möglich, einem Menü einen weiteren Bildschirm zuzuordnen (Zapharis et al. 2002).
Eine Prüfung auf *Deadlocks* erfolgt nach dem Beenden der Modellierung in Form von Reports (vgl. Anhang D). Wenn die Prüfung zur Laufzeit erfolgt, würde bei jedem neuen Anlegen eines Objektes (Bildschirm) ein *Deadlock* entstehen, da die Transition für den Rückweg zu diesem Zeitpunkt noch gar nicht erstellt wurde. So würde die *Deadlock-Prüfung* immer einen Fehler melden, wenn neue Objekte in den Graphen eingebaut werden. Mithilfe von Reports kann der aktuelle Graph auf *Deadlocks* geprüft werden, um so die kritischen Objekte anzeigen zu lassen. Diese können dann grafisch, beispielsweise mit einer roten Umrandung dargestellt werden. Darüber hinaus können den Bediensystementwicklern ergonomische Bausteine in Form von geprüften Interaktionskonzepten bereit gestellt werden. Immer wiederkehrende, vom *Usabiliy-Engineer* spezifizierte Menüabläufe können von den Entwicklern aus einer Bibliothek geholt werden. Somit müssen die Entwickler häufig auftretende Menüabläufe nicht ständig nachmodellieren.

5.4.3 Architektur der Beratungskomponente

Die im Rahmen dieser Arbeit konzipierte Architektur des Beratungswerkzeuges setzt sich aus vier Hauptkomponenten zusammen, dem *GUI-Builder*, dem *CASE-Tool*, dem *Berechnungsframework* und der *Regelbasis* (vgl. Abbildung 5.18). Der GUI-Bilder bietet derzeit keine Möglichkeit an, Plug-ins zu integrieren, da dieser erst seit Anfang März als Vollversion erhältlich ist. Im Konzept dieser Arbeit wird jedoch davon ausgegangen, dass die Firma Microsoft in geraumer Zeit eine Schnittstelle anbietet, um eventuelle Plug-ins zu integrieren.
Die neue MetaEdit+ Version 4.5, welche sich seit Anfang 2007 auf dem Markt befindet,

stellt eine API-Schnittstelle bereit, um auf die Objekte zur Laufzeit zugreifen zu können. In der Regelbasis werden die Metriken und Formeln aus Abschnitt 5.4.1 abgelegt.
Mithilfe des Frameworks kann die Validität auf Basis der Regelformeln berechnet werden. Für das Berechnungsframework bietet sich die Programmiersprache C# in Kombination mit dem NET Framework 3.0 an, da das XAML-Format in Objektklassen der WPF (Windows Presentation Foundation) eingelesen werden kann und diese somit ausgewertet werden können.

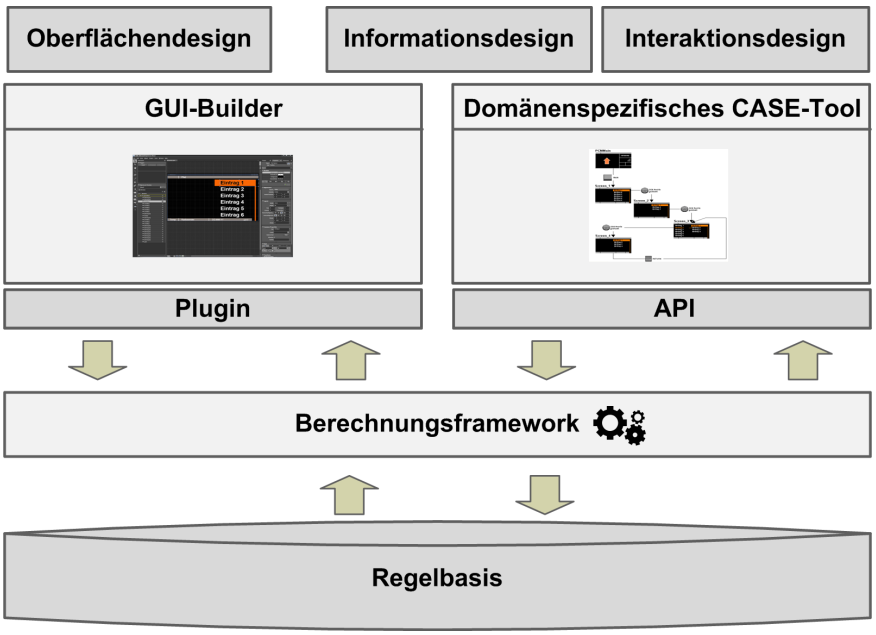

Abbildung 5.18: Konzept Architektur für die Beratungskomponente

Das CASE-Tool bietet umfangreiche Möglichkeiten, Entwickler im Bereich des *Interaktionsdesigns* zu unterstützen. So können die Ein- und Ausgänge der Bildschirme mit Regeln versehen werden. Für die Berechnung der Menütiefe werden die Regeln direkt über die proprietäre Skriptsprache, welche MetaEdit+ zur Verfügung stellt (Reports), abgebildet (vgl. Anhang D). Im nächsten Abschnitt werden die Erkenntnisse aus dem Konzept vorgestellt.

5.4.4 Erkenntnisse aus dem Konzept

Die beschriebenen Konzepte aus Abschnitt 5.4 dienen dazu, den Entwickler während der *Serienentwicklung* zu unterstützen. Ziel hierbei ist es, die möglichen Konzepte aufzuzeigen, die bei der Gestaltung von ergonomischen Bediensystemen eingesetzt werden können, um den Entwickler intuitiv zu unterstützen. Dies ist hauptsächlich dann erforderlich, wenn die Anwendung wie am Beispiel PCM 2.1 der Porsche AG unzählige Dialoge umfasst, die strukturell sehr ähnlich aufgebaut sind.
Der GUI-Builder bietet durch das XAML-Format neue Möglichkeiten für die *Designer* im Entwicklungsprozess. Diese können noch effizienter in den Produktentstehungsprozess integriert werden. Durch den Einsatz des neuen GUI-Builders hat sich die Qualität in der Unterstützung des *Oberflächendesigns* deutlich erhöht. Die Stärke des GUI-Bilders liegt in der intuitiven Bedienung, die dem Bildbearbeitungsprogramm Photoshop ähnelt. So können beispielsweise derzeit schon ergonomische Bausteine in Form von Bildschirmen in der Bi-

bliothek mit Suchfunktion abgelegt werden. Darüber hinaus können derzeit schon Grafiken detailliert gestaltet werden, sowie Animationen eingebunden werden. Dies Funktionen bieten einen erheblichen Mehrwert gegenüber den sich auf dem Markt befindlichen Werkzeugen. Mithilfe der Panels (vgl. Abschnitt 5.4.2) wurde eine mögliche Integration einer Überprüfungskomponente gezeigt. Wenn der GUI-Bilder eine Plug-in Schnittstelle anbietet, könnte mithilfe der Regelbasis eine Überprüfung der Oberfläche stattfinden. Wobei hier sicher abzuwägen, ist auf welche Metriken dann speziell zurückgegriffen wird. Der Zugriff auf die einzelnen Elemente mittels dem Document-Object-Model (DOM) ist anlog zu XML. Das Ablegen der Metriken in der Regelbasis ist derzeit noch mit Programmieraufwand verbunden. Hier könnte beispielsweise ein Dialog-Manager Abhilfe schaffen, wie er beispielsweise im Werkzeug *KWARESMI* (vgl. Abschnitt 4.2.4)eingesetzt wird, um die Regeln für die Evaluations-Sets abzuspeichern.
Im Bereich des *Informationsdesigns* konnte die angedachte Längenprüfung nicht integriert werden, da es zum jetzigen Zeitpunkt nicht möglich ist, die Variablen vom GUI-Builder *(Feldlänge, Schriftart, Schriftgröße, Schriftschnitt)* und Meta-Tool *(Text)* zu vereinen. Alternativ wurde jedoch ein Konzept aufgezeigt, das eine Prüfung durch den XML-Export des kompletten Graphen durchführt. Das Werkzeug MetaEdit+ bietet für den Bereich des *Interaktionsdesigns* sehr viele Möglichkeiten mithilfe des Metamodells *GOPPRR* die Entwickler zu unterstützen.
Weiter ist bei diesem Konzept der Bruch zwischen dem *Oberflächen-* und *Informations-* bzw. *Interaktionsdesign* nicht gelöst. Hierbei könnten eventuell die *Visual Studio DSL-Tools* von Microsoft Abhilfe schaffen, denn mit *Microsoft Expression Blend* können die Grafiken vektorbasiert exportiert werden. Die Vektorgrafiken können vom *Visual Studio DSL-Tool* direkt im XAML-Format importiert werden. Dies hat den Vorteil, dass die Symbole nicht wie bisher im CASE-Tool nachgezogen werden müssen, sobald der *Designer* etwas an der Oberfläche der Bildschirme ändert. Die *Visual Studio DSL-Tool* von Microsoft sind gerade erst auf dem Markt erschienen und scheinen nach Heilig (2007) noch nicht ausgereift genug für den Einsatz in der Werkzeugkette zu sein.

5.5 Konzept des Wissensmanagementsystems

Aufgrund der in Abschnitt 5.2 identifizierten Merkmale wird an dieser Stelle ein Wissensmanagementsystem speziell auf die Anforderungen der Porsche AG entwickelt, um die Qualität des Entwicklungsprozesses zu erhöhen.
Das Modell nach Probst 1998 (vgl. Abschnitt 3.4) identifiziert die wissensbezogenen Aktivitäten sehr treffend, die bei der Ausarbeitung eines Konzepts für ein Wissensmanagementsystem beachtet werden müssen. So muss eine Wissensplattform über eine Reihe von Funktionen verfügen, um die Benutzer effektiv und effizient zu unterstützen. Um beispielsweise das implizite Wissen zu fördern (vgl. Abschnitt 3.4.1), sollte den Entwicklern die Möglichkeit gegeben werden, mit anderen Wissensträgern zu interagieren. Oft lassen sich Probleme durch einen gegenseitigen Austausch dementsprechend lösen. Probst (1998) nennt diesen Aspekt in seinem Modell *Art der Wissensverteilung* (Abschnitt 3.4.1).
Der folgende Abschnitt beschäftigt sich damit, wie eine mögliche Wissensplattform aufgebaut werden könnte. Darüber hinaus wird gezeigt, wie eine Wissensplattform in den aktuellen HMI-Entwicklungsprozess integriert werden kann. Außerdem wird in einem möglichen Anwendungsbeispiel gezeigt, wie das System zum Einsatz bei der Entwicklung von interaktiven Bediensystemen kommt. Der Abschnitt schließt mit den Erkenntnissen, welche aus der Arbeit gewonnen wurden ab.
Abbildung 5.19 zeigt in Anlehnung an Bach et al. (1999) eine vereinfachte Struktur einer Wissensplattform.

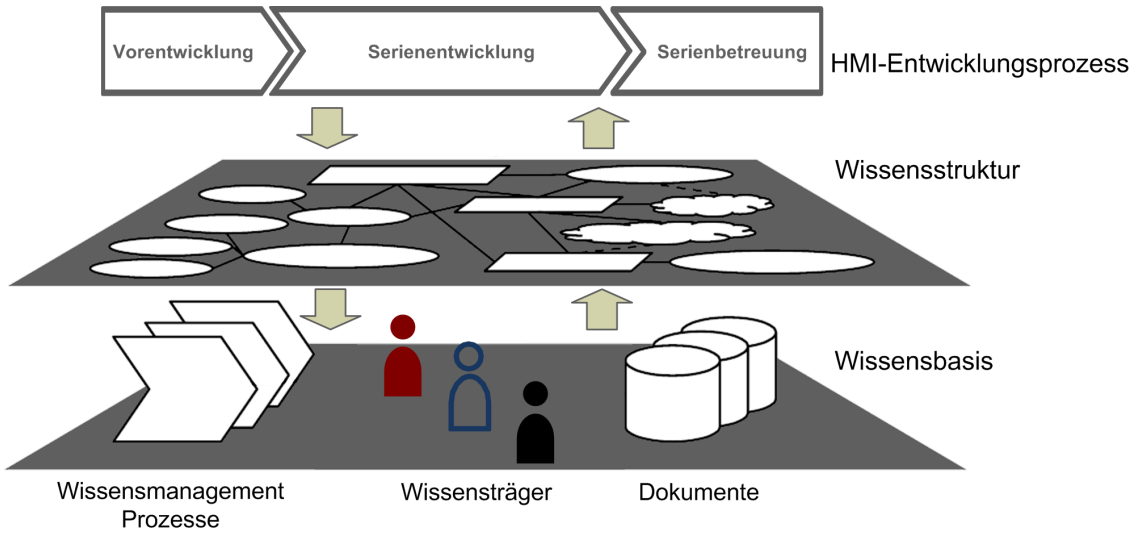

Abbildung 5.19: Vereinfachte Struktur einer Wissensplattform

5.5.1 Systemarchitektur

Wissensmanagementsysteme können die Entwickler aktiv im Umgang mit Wissen unterstützen. Dabei steht das Auffinden von bedeutsamen Informationen durch die Entwickler im Vordergrund. Das Unternehmen Forrester Research unterteilt das Wissensmanagementsystem in *Content*, *Application*, *Data*, *Prozess* und *Collaboration*.
Eine auf dem Forrester Research Ansatz aufgebaute Architektur von Wissensplattformen ist die bekannte Business Knowledge Management-Toolarchitektur (BKM) (Bach et al. 1999, Lehner 2006)(vgl. Abbildung 5.20). Die Inhalte aus unterschiedlichen Quellen werden in sogenannte *Integrationsdienste* zusammengefasst, hierzu zählen beispielsweise Office-Dokumente und Datenbanken. Der nächste Schritt, der sogenannte *Wissensdienst*, ist für die Zugriffe auf die Integrationsdienste verantwortlich. Die Wissensdienste werden mit Funktionalitäten versehen, um so beispielsweise bei einer Suche mithilfe von Metadaten das entsprechende Ergebnis zu erhalten. Die Wissensdienste können wiederum in Lebenszyklus-Dienste und Profil-Dienste unterteilt werden. Zu den Lebenszyklus-Diensten gehört beispielsweise das Verwalten von Workflows. Profil-Dienste bieten hingegen beispielsweise die Suchfunktion. Der sogenannte *integrierte Arbeitsplatz* ist die eigentliche Benutzeroberfläche für die Mitarbeiter. Mithilfe der Komponente kann der Benutzer auf die gewünschten Inhalte des Wissensmanagements zugreifen.

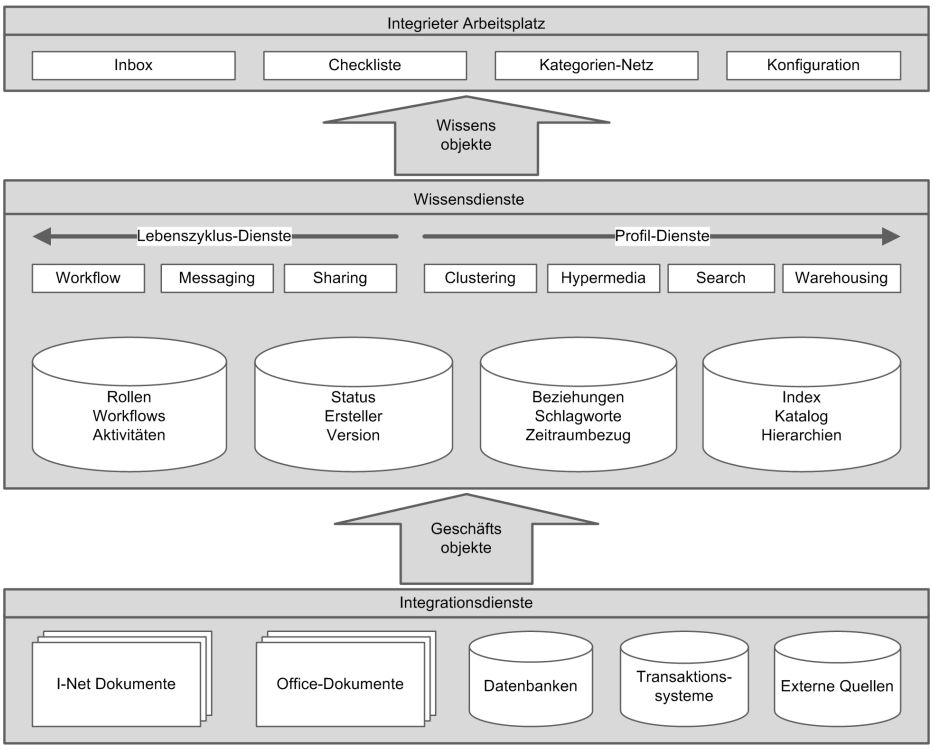

Abbildung 5.20: BKM-Toolarchitektur (Bach und Österle 1999)

Die Herausforderung bei der Erstellung eines Konzepts für die Realisierung eines Wissensmanagementsystems besteht in der Zusammenstellung von Wissensquellen sowie der Integration der zugrunde liegenden Systeme in einer Wissensplattform, so Jansen et al. (2001). Abbildung 5.21 zeigt das erarbeitete Konzept für die Porsche AG.

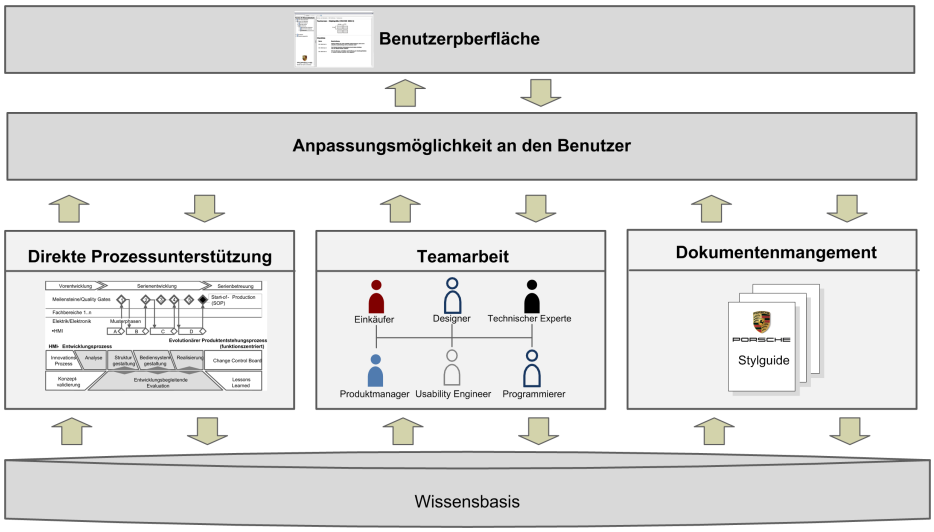

Abbildung 5.21: Aufbau des Porsche Wissensmanagementsystems

Die Konzeption wurde in Anlehnung an die BKM-Toolarchitektur (vgl. Abbildung 5.20) und dem Modell von Jansen et al. (2001) vorgenommen. Die Wissensplattform wurde in *Benutzeroberfläche*, *Anpassungsmöglichkeit an die Benutze*r, *Direkte Prozessunterstützung*, *Teamarbeit*, *Dokumentenmanagement* und *Wissensbasis* aufgeteilt.

Im Anschluss werden die einzelnen Komponenten eingehend erläutert. Zusätzlich werden die Unterstützungsmöglichkeiten aufgezeigt, die dieses System mit sich bringen würden.

Die *Benutzeroberfläche* mit den Navigationselementen ist die Schnittstelle zum Entwickler. Die Oberfläche muss leicht und effizient mit vertrauter Notation zu bedienen sein, um so einen schnellen und effizienten Zugriff auf die gewünschten Informationen zu ermöglichen. Dies ist ein wichtiges Kriterium, um die Barriere zum Wissensmanagementsystem abzubauen (vgl. Abschnitt 3.4.3). Die Entwickler müssen an jeder Stelle des Systems über die jetzige Position im Kontext informiert sein. Darüber hinaus muss die Eingabe von neuem Wissen nach einem genauen Muster erfolgen, um so die Konsistenz des Systems zu gewährleisten (Studer et a. 2001). Der Benutzter muss bei der Eingabe neuer Informationen geführt werden.

Die *Anpassungsmöglichkeit an den Benutzer* erlaubt beispielsweise das individuelle Konfigurieren des Arbeitsplatzes. Besonders interessante Inhalte können als Favoriten abgespeichert werden, um so Verweise auf die Informationen zu haben. Dies ist für die *Usability-Engineers* von großem Vorteil, so können sie beispielsweise die Normen und Standards als Favoriten hinterlegen, um einen schnellen Zugriff auf die Dokumente zu haben. Eine persönliche Inbox könnte für persönliche Meldungen allerart bereitgestellt werden.

Zur direkten *Prozessunterstützung* werden basierend auf der Wissensbasis hinterlegte Vorgehensmodelle bereitgestellt. Zu den Modellen zählen beispielsweise einfache Checklisten bis hin zu umfassenden Workflows. Diese Komponente soll sicherstellen, dass alle am Entwick-

lungsprozess beteiligten Personen entlang den Arbeitsabläufen mit dem benötigten Wissen versorgt werden. Da die Entwicklung eines Bediensystems ein relativ strukturierter Ablauf ist, kann diese Form der Unterstützung die Qualität des Produktes deutlich erhöhen. Mithilfe von Aufgabenlisten, welche in einer bestimmten Reihenfolge bearbeitet werden müssen, kann die Transparenz im Entwicklungsprozess erhöht werden. Zum einen weiß der Bearbeiter genau, was er noch umzusetzen hat, zum anderen sehen die restlichen Entwickler, wie weit die einzelnen Personen bei der Bearbeitung ihrer Aufgaben sind, da vorstellbar wäre, diese direkt mit dem Projektplan zu synchronisieren.
Der größte Teil des Wissens steckt in den Köpfen der Entwickler und ist nicht explizit dokumentiert (vgl. Abschnitt 3.4). Somit ist die *Teamarbeit* bei einem Wissensmanagementsystem unumgänglich. Die Entwickler verfügen über das nötige Know-how, um ein Bediensystem erfolgreich zu entwickeln. Deshalb sollte die Wissensdatenbank eine Komponente beinhalten, die die Unterstützung von Gruppenarbeit fördert. Idealerweise steht den Entwicklern eine Art Diskussionsforum zur Verfügung. Das Forum bietet eine schnelle und informelle Klärung von Fragen und Problemstellungen. So können die Entwickler gegebenenfalls eventuelle Änderungen beim Projektverlauf schnell und effektiv kommunizieren. Kuhlen (2000) nennt diese Form von Kommunikation *Koordinationsfunktionen*. Eine weitere Möglichkeit, die Entwickler zu unterstützen, ist eine Expertensuche. Shneiderman (1998) nennt diese Art von Unterstützung *Hilfenetzwerke* (vgl. Abschnitt 3.3.3). Die Mitarbeiter der Porsche AG legen von sich ein Profil an, in dem der Tätigkeitsbereich beschrieben wird. Mithilfe der Expertensuche könnten Spezialisten ausfindig gemacht werden, wenn zu bestimmten Themenbereichen Probleme auftreten.
Ein *Dokumentenmanagementsystem* ermöglicht den Entwicklern die Ablage von Dokumenten in einer Art Verzeichnisbaum. Außerdem sollte auch das Ablegen von Grafiken, Animationen und Videos möglich sein. Animationen wirken sich positiv auf das Lernen und vor allem auf das Behalten von Texten aus (Shneiderman 1998) (vgl. Abschnitt 3.3.3). Der effiziente Zugriff auf die Informationen ist daher genau so wichtig, wie die Weiterbearbeitung. Zusätzlich sollte das System die Vergabe von Zugriffsrechten (Rollenkonzept) sowie die Speicherung von Metainformationen zulassen. Für *Lotus Domino* ist eine Dokumenten Management Software erhältlich. Dies hat den Vorteil, dass auf die derzeit verwendete Software der Porsche AG, *Lotus Notes*, zurückgegriffen werden könnte. Der *Lotus Notes Domino Document Manager 7* speichert die Dokumente in einer Bibliothek ab. Die offenen Anwendungsschnittstellen (API) ermöglichen mit der IBM *Lotus Notes* Skriptsprache, Microsoft Visual Basis und Visual C++ den Zugriff auf alle Funktionen[7].
Darüber hinaus wäre es denkbar, den Entwicklern die Möglichkeit zu bieten, interessante Themengebiete zu abonnieren, d.h wird eine Änderung im Dokument vorgenommen, so werden die Entwickler automatisch benachrichtigt. Diese Benachrichtigung kann in Form einer E-Mail oder einer Meldung in der persönlichen Inbox durchgeführt werden. Jeder Entwickler kann Dokumente anlegen, löschen und verändern. Diese Vorgänge dürfen aber nur im zuvor definierten autorisierten Bereich vorgenommen werden. Ferner kann über eine Historienliste aufgeführt werden, welcher Autor das Dokument bereitgestellt bzw. verändert hat.

[7]Datenblatt von Lotus Notes Domino Document Manager
(ftp://ftp.software.ibm.com/software/emea/de/lotus/G325_2044_03_GE.pdf) Letzter Zugriff: 03.08.2007

Mit einer Versionierungsfunktion wird die Nachvollziehbarkeit von eventuellen Änderungen deutlich gemacht. Das gemeinsame Arbeiten an Dokumenten und der zeitgleiche Zugriff wird vom Dokumenten-Manager (z.B. *Lotus Notes Document-Manager 7*) bereitgestellt. Pflegt ein Entwickler ein neues Dokument ein, so erfolgt die automatische Einordnung des neuen Dokumentes anhand der eingebenden Parameter.
Die Konvertierung in Standardformate wie beispielsweise doc, pdf und XML muss gewährleistet sein. Ferner muss es möglich sein, dass die Entwickler eine Bewertung des Inhalts durchführen können. Diese Bewertung kann aus Erfahrungswerten von anderen Projekten stammen. Das Wissensmanagementsystem sollte eine Volltextsuche und Navigation entlang von Baumstrukturen ermöglichen. Zudem sollte die Suche von Informationen über Metadaten durchgeführt werden können. Darüber hinaus ist die Bereitstellung von Wissen gegliedert in die Projektphasen, die einen deutlichen Mehrwert für die Entwickler mit sich bringen. Sucht der *technische Experte* beispielsweise Normen zur Bediensystemgestaltung, so kann er diese nicht nur in der Ordnerstruktur unter *Normen* und *Standards* finden, sondern auch im Verzeichnis *Serienentwicklung*, *Bediensystemgestaltung*, *Normen*.
In der *Wissensbasis* werden die Wissensquellen und die Informationen über die Struktur abgelegt. Die Wissensbasis ist vergleichbar mit einer Datenbank, in der sämtliche Daten und Dokumente enthalten sind.

5.5.2 Integration des Konzepts in den Entwicklungsprozess

Die Integration der Wissenskomponente wird wie im vorherigen Abschnitt anhand der einzelnen Projektphasen erläutert. Somit können die verschiedenen Wissensinhalte in den einzelnen Projektphasen angeboten werden. Die Eindämmung der Informationsflut ist von großer Bedeutung (Delphi Group1998). Dies kann auch auf den HMI-Entwicklungsprozess übertragen werden, denn nicht jede Person benötigt entsprechend gleich viel an Informationen. In den Phasen der *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung* wird den Entwicklern spezielles Wissen zu den einzelnen Projektphasen bereitgestellt.

Abbildung 5.22: Die Porsche AG Wissenspyramide

Darüber hinaus wird auch Wissen bereitgestellt, welches für die Bearbeitung des aktuellen

Projektes benötigt wird. Zusätzlich wird den Entwicklern auch projektübergreifendes Wissen angeboten. So kann beispielsweise mithilfe dieses Wissens eine bereichsübergreifende Kooperation geschaffen werden. Aus diesen drei Wissensbereichen wurde eigens für diese Arbeit eine *Wissenspyramide* entwickelt.
Die folgende Abbildung zeigt die Wissenspyramide, eingeteilt in die verschiedenen Schichten des Wissens, die während des Entwicklungsprozesses die Entwickler unterstützen.
In Folgenden werden die verschiedenen Schichten vorgestellt, die Informationen für die Entwickler anbieten. Wie stark die verschiedenen Entwickler in den einzelnen Phasen eingebunden sind, wurde bereits in Abschnitt 5.2 erläutert.

Wissen in den Projektphasen

Wie schon im obigen Abschnitt erläutert, ist das Bündeln von Informationen sehr wichtig, um die Entwickler bei der Suche nicht zu überlasten. Im Folgenden werden die Informationen, welche in den Projektphasen angeboten werden, näher beschrieben.

Vorentwicklung Im Bereich der *Vorentwicklung* können die Entwickler auf *Lessons Learned* aus früheren Projekten zugreifen. So finden sie eine Sammlung von bekannten Problemen und Risiken, die zuvor während der Entwicklung aufgetreten waren. Ferner können die *Designer* und *technischen Experten* auf einschlägige Normen und Standards zurückgreifen, die schon während der *Vorentwicklung* zu beachten sind. So gilt es hier schon abzuwägen, ob es beispielsweise sinnvoll ist, ein Display mit Touchscreen im Fahrzeug einzusetzen. Mithilfe der angebotenen einschlägigen Normen haben die Entwickler die Möglichkeit, schon in sehr frühen Phasen des Entwicklungsprozesses eventuelle Einschränkungen des Konzepts zu erkennen.
Der *Produktmanager* führt ständig Marktbeobachtungen durch. Diese kann er problemlos über das Wissensmanagementsystem publizieren. Somit können sich die Entwickler in der Phase der *Vorentwicklung* durch Konkurrenzprodukte inspirieren lassen. Die Entwickler können sich so auf ihre Domäne konzentrieren und müssen nicht ihre Zeit mit aufwendigen Recherchen verbringen.

Serienentwicklung In dieser Entwicklungsphase haben die am Prozess beteiligten Personen ebenfalls die Normen und Standards abzurufen. Hierbei ist es sehr wichtig, dass die Entwickler direkt aus der Werkzeugkette (vgl. Abschnitt 3.3.2)auf die Wissenskomponente zugreifen können, ohne eine andere Anwendung starten zu müssen. Darüber hinaus können die Entwickler auch in dieser Phase mithilfe der Unterstützungskomponente schnell und effizient bei eventuell auftretenden Fragen kommunizieren.
Zusätzlich wurde im Rahmen dieser Phase ein Textmanagement-Tool konzipiert. Mithilfe dieser Anwendung können die Entwickler die Beschriftungen für das Bediensystem verwalten (vgl. Abbildung 5.23). Die eingetragen Beschriftungen, welche auch über die API des MetaCASE-Tools zu Verfügung gestellt werden können, sind von einer verantwortlichen Person verifiziert worden und dürfen somit verwendet werden. In der ersten Spalte werden die zu entwickelnden Systeme aufgeführt. In der nächsten stehen die Textgruppen, wie beispielsweise für das PCM 2.1: *Bedienelemente*, *Menüeinträge* und *sonstige*. Im Beispiel wurde die

Textgruppe *Bedienelemente* ausgewählt. Die dazu gehörigen Beschriftungen erscheinen in der dritten Spalte. Wählt nun der Entwickler das Bedienelement *Audio* aus, so erscheint in der letzten Spalte die dazugehörige Übersetzung. Das Textmanagement-Tool läuft in einer eigenen Anwendung, kann aber jeder Zeit über das Wissensmanagementsystem gestartet werden. Durch Modularisierung könnte das System auch in anderen Projekten des Porsche Konzerns benutzt werden, so auch beispielsweise bei der Verwaltung von Beschriftungen für Bedienknöpfe im Interieur des Fahrzeuges.

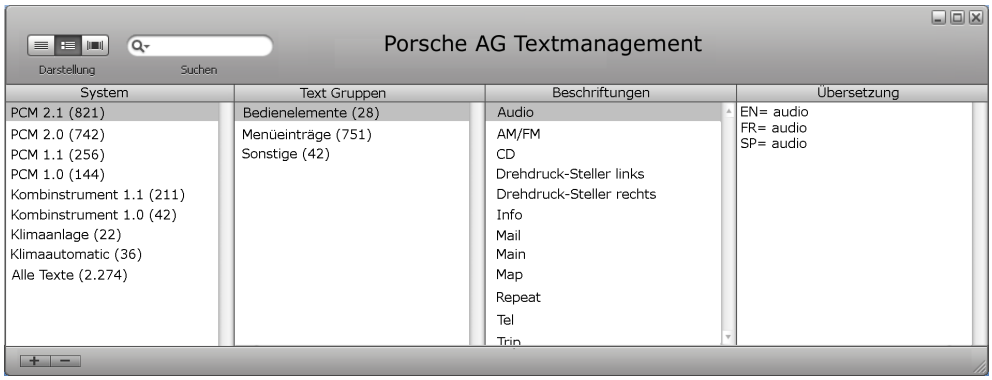

Abbildung 5.23: Mockup Textmanagement-Tool

Serienbetreuung In der *Serienbetreuung* werden Informationen eher in das System eingeben als beispielsweise auftretende Fehler auf dem *Change Control Board*. Um die Qualität der Produkte weiter zu erhöhen, könnten die zertifizierten Porsche Vertragshändler mit dem Wissensmanagementsystem vernetzt werden, um so entsprechende Mängel direkt an den verantwortlichen Entwickler zu adressieren. Die Nachricht kommt bei dem verantwortlichen Entwickler in der persönlichen Inbox an. Dies kann mit *Lotus Notes* problemlos durchgeführt werden. Von der Maßnahme, dass die bisherigen Erfahrungen und Fehler transparent in einem System abgelegt werden können, profitieren die Entwickler sehr stark in der *Vorentwicklung*. Ferner können die Entwickler jederzeit auf Normen und Erfahrungen zurückgreifen.

Wissen im Projekt

Im vorherigen Abschnitt wurde der Einsatz des Wissensmanagementsystems in den einzelnen Projektphasen der *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung* vorgestellt. In diesem Abschnitt werden die Informationen behandelt, die nicht direkt einer Phase zuzuordnen sind. So können hier Erfahrungen abgelegt bzw. abgerufen werden, die die Schnittstellen zwischen den einzelnen Phasen näher betrachten. Darüber hinaus kann der Projektplan für die aktuelle Entwicklung transparent dargestellt werden. Dieser kann mit Kommentaren der Entwickler versehen werden, um so beispielsweise das Vorankommen bei der Bediensystemgestaltung transparent zu gestalten. Diese Funktion soll keine Überwachung der Mitarbeiter (vgl. Abschnitt 3.4.3) sein, sondern das kollektive und kollaborative Arbeiten am Produkt beschleunigen.
Ferner werden die Ergebnisse der Entwickler in einer Art Produkt-Styleguide festgehalten (vgl. Abschnitt 3.1.1). Der Porsche Produkt-Styleguide ist ein vorgefertigtes elektronisches

Dokument, in dem alle Ergebnisse der Entwicklung transparent in definierte Muster abgelegt werden können. Das gleichzeitige Arbeiten am Produkt-Styleguide muss gewährleistet sein. So gehen während des HMI-Entwicklungsprozesses keine Informationen oder Ergebnisse verloren. Darüber hinaus bietet der Porsche Produkt-Styleguide ein bestimmtes Qualitätsmaß durch die Festlegung eines Standards. Die Ergebnisse können bei Folgeprojekten wieder in den Prozess mit einfließen.
Außerdem können Arbeitsplatzbeschreibungen (vgl. Abschnitt 3.4.1) der einzelnen Tätigkeitsfelder abgelegt werden, um so neue Mitarbeiter, die nur für dieses Projekt hinzugezogen werden, einzulernen. So muss kein zusätzlicher Mitarbeiter für die Einführung des neuen Kollegen abgestellt werden. Neben den Arbeitsplatzbeschreibungen können auch Schulungsunterlagen bereitgestellt werden. So können sich die Entwickler individuell anhand der Unterlagen weiterbilden. Denkbar hierbei ist es, die Arbeitsschritte in Tutorials zu fassen (vgl. Abschnitt 3.3.3). So können die Tätigkeiten Schritt für Schritt vermittelt werden. Als Beispiel hierfür kann der Einbau eines Prototypen von einem Fahrerinformationssystem angeführt werden, bei dem einem Entwickler eine detaillierte Anleitung gegeben wird.

Projektübergreifendes Wissen

Das projektübergreifende Wissen bezieht sich auf Informationen, die nicht den Projektkontext betreffen, sondern den Prozess als Ganzes. Hierbei können neben den Erfahrungen im Entwicklungsprozess Verbesserungsvorschläge von Mitarbeitern aufgenommen werden (vgl. Abschnitt 5.2.4).
Diese senden beispielsweise die Verbesserungsvorschläge, die sie während der Durchführung eines Projekts beobachtet haben, in einem standardisierten elektronischen Formular an den Abteilungsleiter. Dieser ist über die einzelnen Prozessphasen genauestens informiert und kann gegebenenfalls die Verbesserungsvorschläge des Mitarbeiters im nächsten Meeting ansprechen. Erkennt der Abteilungsleiter sofort, dass der Verbesserungsvorschlag nicht durchsetztbar ist, so sendet er dem jeweiligen Mitarbeiter, der den Vorschlag eingereicht hat, eine Ablehnung mit der entsprechenden Begründung zu. Somit erhält der Porsche Mitarbeiter Feedback und wird darüber informiert, weshalb sein Vorschlag nicht umgesetzt werden kann. Gleiches gilt bei der Annahme des Verbesserungsvorschlages, der Mitarbeiter erhält einen Nachricht.

5.5.3 Zugriff auf das Wissensmanagementsystem

Im Porsche Konzern ist das dokumentorientierte Datenbanksystem *Lotus Notes* im Einsatz. Das Wissensmanagementsystem kann darin problemlos integriert werden. Jeder Mitarbeiter kann auf diese Anwendung von seinem Arbeitsplatz aus zugreifen.
Selbst bei Testfahrten im Ausland können sich die Entwickler die aktuellen Informationen von der Datenbank replizieren. Zusätzlich ist so der problemlose Zugriff auf die Mailaccounts der Mitarbeiter der Porsche AG gegeben. So kann beispielsweise eine Weiterleitungsadresse mühelos konfiguriert werden.

Zugriff aus Lotus Notes

Der Zugriff auf das Wissensmanagementsystem erfolgt über eine sogenannte *Lotus Notes Kachel* die sich nach dem Einloggen in das System auf dem Startbildschirm befindet. Je nach Freischaltung der Wissensdatenbank haben die Entwickler Zugriff auf die Informationen. Wie in Abschnitt 3.3.2 ausführlich beschrieben, wird so die Hilfe über *Lotus Notes* passiv, also durch explizite Anfrage der Entwickler gestartet. Darüber hinaus ist die Unterstützung auch zu Beginn statisch, da der aktuelle Kontext nicht berücksichtigt wurde. Dies ändert sich aber, wenn der Benutzer beispielsweise beim Recherchieren nach speziellen Normen über die Suche der *Wissenspyramide* geht. Dabei werden ihm nur die Normen angezeigt, die für diese Prozessphase wichtig sind. Die Form der Suche ist problemorientiert. Je mehr der Entwickler über das zu suchende Objekt weiß, desto stärker kann er die Suchkriterien eingrenzen. Diese Form der Suche lehnt sich an das Werkzeug GUIDE an. Dort können die Entwickler durch das Eingrenzen der Problemstellung schneller auf spezifische Informationen zugreifen (vgl. Abschnitt 4.3.4).

Zugriff aus der Werkzeugkette

Der Zugriff auf das Wissensmanagementsystem muss auch aus den Werkzeugen der Werkzeugkette heraus möglich sein. So könnte beispielsweise durch Drücken der F1-Taste das Wissensmanagementsystem automatisch starten. Beim Starten werden automatisch Themengebiete der *Serienentwicklung* geladen (vgl. Abschnitt 3.3.2), weil in dieser Phase vornehmlich die Werkzeugkette eingesetzt wird. Der *Microsoft Expression Blend* verfügt derzeit über keine Schnittstelle, um den Zugriff auf die Wissensdatenbank *(Lotus Notes Domino)* zu realisieren. Es sollte auch beim GUI-Builder möglich sein, mit einem Tastendruck die Wissensplattform zu starten, um den Kontextbezug (vgl. Abschnitt 3.3.2) zu den abgelegten Informationen herzustellen. Das MetaCASE-Tool verfügt hingegen über eine API, über die der Zugriff auf die *Lotus Notes Domino* Datenbank durchgeführt werden kann. Mithilfe der IMB *Lotus Notes* Skriptsprache ist es möglich, auf die *Lotus Notes* API zuzugreifen. So können die Daten aus der Datenbank ausgelesen und in die Werkzeugkette integriert werden.

5.5.4 Anwendungsbeispiel

In diesem Abschnitt wird ein Anwendungsbeispiel des Wissensmanagementsystems vorgestellt. Dabei wird anhand eines Mockups erläutert, wie das System, welches in den vorherigen Abschnitten konzipiert wurde, die Entwickler unterstützt. Zu Beginn wird ausführlich die Benutzeroberfläche des Systems erläutert. Im Anschluss wird gezeigt, wie die am Entwicklungsprozess beteiligten Personen ein neues Dokument in die Wissensplattform einpflegten können.

Einführung Benutzeroberfläche

Die Benutzeroberfläche des Wissensmanagementsystems wurde in verschiedene Bereiche aufgeteilt. An der linken Seite befindet sich der *Gliederungsbereich*, die *Persönliche Inbox*, und

die *Gruppendiskussionsfunktion*. Im oberen Teil des Fensters befindet sich die *Navigationsleiste*. In der Mitte ist der Bereich, in dem beispielsweise die Normen angezeigt werden können bzw. dargestellt sind. Im Folgenden werden die einzelnen Komponenten näher erläutert.

Mithilfe der *Navigationsleiste*, welche sich im oberen Bereich des Systems befindet, können die Entwickler auf die *Wissenspyramide*, den *Projektplan*, die *Favoriten*, *Neues Dokument*, *Kommentar* und das *Textmanagement-Tool* zugreifen. Darüber hinaus befindet sich die *Logout-Funktion* am rechten Rand der Navigationsleiste (vgl. Abbildung 5.24).

Abbildung 5.24: Benutzeroberfläche Wissensmanagementsystem

Wissenspyramide: Mit dieser Funktion können sich die Entwickler die *Wissenspyramide*, welche in Abschnitt 5.5.2 eingehen erläutert wurde, anzeigen lassen. Somit haben sie einen schnellen und effektiven Zugriff auf die Informationen. Der Vorteil dieser Suchfunktion liegt darin, dass die Entwickler schon eine vorgefertigte Strukturierung der Daten vorfinden. Wie aus den Interviews hervorging, hatten die Entwickler beispielsweise Probleme die große Anzahl an Normen und Standards bei einer gezielten Suche zu bewältigen. Durch diese Funktion werden die Informationen nach den Projektphasen, das Wissen im Projekt und dem projektübergreifenden Wissen unterteilt. Sucht ein Entwickler nach speziellen Normen im Bereich *Zulässige Farbgebung für Fahrerinformationssysteme*, so muss dieser nur in der *Wissenspyramide* auf die Normen und Standards klicken, die im Bereich der *Serienentwicklung* angeboten werden. Nun erhält der Entwickler eine kategorisierte Anzeige der Normen in *Oberflächen-*, *Informations-* und *Interaktionsdesign*. Da es sich bei der Farbgebung um den Bereich des *Oberflächendesigns* handelt, kann er seine Suche weiter einschränken, indem er Normen und Standards zum *Oberflächendesign* auswählt. Durch das hybride Bedienkonzept werden am linken Rand die Gliederungsbereiche ständig angepasst, sodass der Entwickler ein Orientierung hat, an welcher Stelle er sich gerade befindet.

Projektplan: Mit dieser Funktion können die Entwickler auf aktuelle Projektpläne zugreifen. Diese müssen aber erst von einer autorisierten Person freigeschalten werden. Im Projektplan sind auch die aktuellen Änderungen eingetragen sowie Kommentare der einzelnen Entwickler bei eventuellen Diskrepanzen.
Favoriten: Diese Funktion stellt spezielle zuvor definierte Links zu Verfügung, um so einen schnellen zugriff auf wichtige Themengebiete zu haben. Diese sind vergleichbar mit den Favoriten bei einem Web-Browser. Hat sich ein Entwickler beispielsweise den Favoriten *Menüstruktur für PCM 2.1* angelegt, so bekommt er eine Nachricht in seine persönliche Inbox, wenn sich bei diesem Eintrag etwas verändert. Grund kann beispielsweise sein, dass der *Usability-Enginner* einen Kommentar angelegt hat, weil er die Menüstruktur mit Probanden getestet hat und diese nicht zurecht kamen. Somit haben die Entwickler bei der Definition von Favoriten auch die Info, wenn sich entsprechend am Datensatz etwas ändert.
Neues Dokument: Die Entwickler können mit diesem Punkt ein neues Dokument im Wissensmanagementsystem anlegen. Wie das genaue Anlegen funktioniert, wird im nächsten Abschnitt umfassend beschrieben.
Kommentar: Zu jeder Information, die im Wissensmanagementsystem bereitgestellt wird, können die Entwickler ihren Kommentar abgeben. Diese Funktion ist wichtig, um von erfahrenen Entwicklern zu lernen.
Textmangement-Tool: Ermöglicht den schnellen Zugriff auf das Textmangement-Tool.

Im *Gliederungsbereich* liegen die eingepflegten Daten in Form einer Ordnerstruktur vor. Darüber hinaus können diese nach Inhalt und Index angezeigt werden. Die Anordnung ist nicht alphabetisch. Die Entwickler können diese frei wählen, um so die wichtigen Themenbereiche nach oben zu setzten. Eine Suchfunktion, die auch auf Metadaten angewendet werden kann, rundet diesen Bereich ab.
Die *persönliche Inbox* ist mit einem Postfach in einem E-Mail Programm zu vergleichen. In dieser können Aufgaben, Nachrichten und Händleranfragen angezeigt werden. Die noch zu bearbeitenden Aufgaben werden in einer TODO-Liste dargestellt. Dies kann auch von einem anderen Entwickler über das System zugesendet werden, um so den reibungslosen Ablauf der Entwicklung zu gewährleisten. Wie oben beschrieben können Dokumente in Form von *Favoriten* abonniert werden. Ändert sich in diesen etwas, so wird dies in der Inbox angezeigt. Spezielle Händler-Anfragen zum Themengebiet kommen auch an der Inbox an.
Das *Gruppendiskussion-Fenster* befindet sich in der linken unteren Ecke. Mithilfe dieses Systems können die Mitarbeiter der Porsche AG, ähnlich wie mit einem Instant Messenger miteinander kommunizieren. Es ist auch über Konferenzschaltung möglich, mehrere Teilnehmer einzuladen.
Im mittleren Bereich werden die Information dargestellt. Im nächsten Abschnitt wird gezeigt, wie ein neues Dokument in die Wissensdatenbank eingepflegt werden kann.

Einpflegen des Styleguides

Mithilfe des Rollenkonzeptes wird das Einpflegen von neuem Wissen in die Datenbank gesteuert. So kann beispielsweise der *Usability-Engineer* nach dem Einloggen in das System einen neuen Styleguide einpflegen (vgl. Abbildung 5.25). Bei Änderungen an der Wissensda-

tenbank müssen sich die Mitarbeiter der Porsche AG einloggen. Der Name und das Datum der Änderung wird ebenfalls dokumentiert.

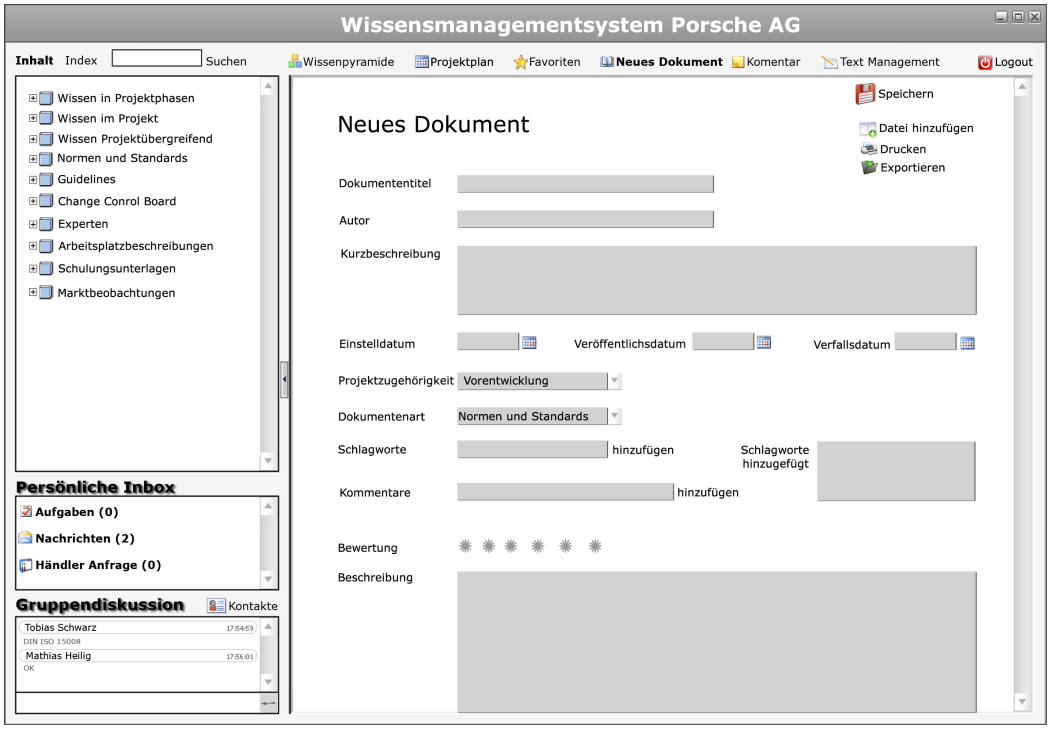

Abbildung 5.25: Einpflegen Styleguide in die Wissensplattform

Zu Beginn wird der Dokumententitel und der Autor in das System eingegeben. Im Anschluss folgt ähnlich wie bei *ProUSE* (vgl. Abschnitt 4.1.5) eine Kurzbeschreibung des Dokumentes. Ferner kann vom Usability-Engineer das Veröffentlich -und Verfallsdatum angegeben werden. Das Einstelldatum wird vom System gesetzt. Im nächsten Schritt müssen über eine Drop Down Liste die Projektzugehörigkeit und die Dokumentenart ausgewählt werden. Beim nächsten Punkt können Schlagwörter bzw. zusätzliche Metadaten hinzugefügt werden. Wobei diese Funktion auch im Lesemodus bereitgestellt wird, d.h die Mitarbeiter können zusätzlich Schlagwörter hinzufügen. Diese erscheinen rechts daneben. Die Dokumente können außerdem durch Anwählen von Sternen bewertet werden. Zum Abschluss erfolgt die eigentliche Beschreibung des Dokumentes. Dies kann zusätzlich mit Bildern, Animationen, Filmen usw. durch Hinzufügen von Dateien angereichert werden. Der eingebene Styleguide kann jederzeit gedruckt oder in Form von einem doc oder pdf exportiert werden.

5.5.5 Erkenntnisse aus dem Konzept

Im vorherigen Abschnitt wurde ein auf die Porsche AG zugeschnittenes Konzept vorgestellt. Denn jedes Unternehmen, das eine Wissensplattform einführt, muss aus den verschieden Möglichkeiten eine individuelle Lösung finden, da es kein Patentrezept gibt, womit sich das Wissensmanagement innerhalb kürzester Zeit etablieren lässt (Schmid 1999). Dennoch gibt es einige Referenzen, wie beispielsweise die BKM-Architekturen (vgl. Abschnitt 5.5.1),an denen sich Unternehmen bei der Einführung eines Wissensmanagementsystems orientieren können. Diese Architektur diente als Anknüpfungspunkt für den Aufbau des Systems, wel-

ches im Rahmen dieser Arbeit konzipiert wurde.
Darüber hinaus wird derzeit beim Porsche Konzern *Lotus Notes* eingesetzt. In dieses System kann problemlos eine Art Dokumenten-Manager integriert werden. Derzeit gibt es bereits einen Ansatz der Firma IBM *(Lotus Domino Document Manager 7)*, wie ein Dokumentenmanagementsystem in *Lotus Notes* integriert werden kann. Basierend auf dieser Anwendung, welche eine umfassende API bereitstellt, könnte ein Wissensmanagementsystem angedacht werden. Somit ist das Replizieren von Wissen standortunabhängig.
Den Entwicklern der Porsche AG wird eine persönliche Wissensumgebung bereitgestellt, die sie individuell konfigurieren können (*Favoriten, Aufgabenlisten*). Mithilfe der entwickelten *Wissenspyramide* kann die große Anzahl der Informationen sinnvoll strukturiert werden. Die Wissensplattform dient auch als Kommunikationsinstrument zwischen den einzelnen Disziplinen im HMI-Entwicklungsprozess. Der Austausch von Informationen unter den interdisziplinären Teams fördert die Qualität des Produktes. Darüber hinaus würde das konzipierte *Textmanagement-Tool* die Arbeit der Entwickler enorm erleichtern. Dies würde die Konsistenz der Texte für die Prototypen sicherstellen. Außerdem werden beispielsweise die Menüeinträge in verschiedenen Landessprachen angeboten.
Jedoch müssen die möglichen Barrieren, die in Kapitel 3 umfassend beschrieben wurden, berücksichtigt werden. Die Sensibilisierung der Mitarbeiter für das Thema Wissensmanagement ist auch Aufgabe des Managements. Um die konsistente Einhaltung der Wissensstruktur zu gewährleisten, d.h., dass auch beispielsweise der neu eingegebene Styleguide in der entsprechenden Kategorie zu finden ist, sollte unter den Mitarbeitern eine Art Wissensmanager bestimmt werden. Dieser ist für die Konfiguration des Systems verantwortlich.
Darüber hinaus beschreibt Sauter (1999), dass Wissensmanagementsysteme in den Unternehmen niemals einen definierten Endzustand erreichen werden. Somit müssen die gespeicherten Inhalte in der Wissensplattform von Zeit zu Zeit kritisch hinterfragt werden, um so die Qualität des Wissens zu verbessern.

5.6 Mehrwert der Unterstützungskomponenten

Der Mehrwert der Werkzeugkette wird mittels der Kriterien des DATech-Prüfhandbuches Usability Engineering-Prozess bewertet, um so ein greifbares und zugleich qualitatives Ergebnis zu bekommen (vgl. Abschnitt 3.2.1). In der Arbeit von Heilig (2007) wurde die Nützlichkeit der Werkzeugkette ohne die Unterstützungskomponente untersucht. Aufbauend auf der Untersuchung von Heilig (2007) wird der Mehrwert der Unterstützungskomponente eingehend erläutert. Darüber hinaus werden abschließend die Erkenntnisse aus der Arbeit vorgestellt.

5.6.1 Prüfung der Kriterien nach DATech

In diesem Abschnitt werden alle 15 Beurteilungsdimensionen betrachtet. Es wird aber auf eine ausführliche Erläuterung der einzelnen Indikatoren der Beurteilungsdimensionen verzichtet. Die genaue Beschreibung der einzelnen Beurteilungsdimensionen und deren Indikatoren können im DATech-Prüfhandbuch (2006) zum Usability Engineering-Prozess nachgelesen werden. Im Rahmen dieser Arbeit wird das Prüfkriterium nur kurz beschrieben. Der Fo-

kus liegt auf der Analyse des Mehrwerts, der durch die Unterstützungskomponente erreicht werden kann.

Usability-Qualitätsziele

Anhand der ersten Dimension *(Beurteilungsdimension 1)* werden die Qualitätsmerkmale, welche im Prozess verwendet werden sollten, wie beispielsweise ein Software-Qualitätshandbuch der Herstellerorganisation, festgelegt. Ohne die Unterstützungskomponente wurde der Reifegrad von Heilig (2007) mit *1* bewertet. Mithilfe der Wissensdatenbank können alle Indikatoren bis auf das *Kriterium 1.6* (Vereinbarung mit Kunden über Usability Normen getroffen) der *Reifestufe 2* erfüllt werden. So wird beispielsweise vom *Indikator 1.4* gefordert, dass in einem Art Qualitätshandbuch die Aktivitäten und Meilensteine zur Erreichung qualitativer Ergebnisse festgeschrieben sind. Dieses Kriterium kann durch den transparenten Projektplan, der vom Wissensmanagementsystem bereitgestellt wird, erfüllt werden. *Indikator 1.5* wird ebenfalls erfüllt, da eine Dokumentation der Usability-Normen in die Wissensplattform integriert werden kann. Schließlich reicht dies aber nicht aus, um *Reifestufe* 2 zu erreichen, da dafür alle Indikatoren erfüllt sein müssen.

Benutzerbeteiligung

Diese Dimension *(Beurteilungsdimension 2.1)* untersucht die Benutzerbeteiligung im Entwicklungsprozess. Nach der Untersuchung des Prozesses ohne die Unterstützungskomponente wurde die Beurteilungsdimension von Heilig (2007) mit der *Reifstufe 0* bewertet. Die unterstützende Komponente könnte den Nutzungskontext, welcher bei der Analyse evaluiert wurde, transparent für Entwickler abspeichern, oder aber durch Bereitstellung von standardisierten Fragebögen die Entwickler bei der Analyse des Nutzungskontextes unterstützen. Mithilfe der Komponente können jedoch die zukünftigen Benutzer nicht besser integriert werden. Das Problem der frühen Nutzerintegration liegt am HMI-Entwicklungsprozess selbst. Das bedeutet, die Reifestufe kann an diesem Punkt nicht erhöht werden.

Requirements-Engineer

Durch die ständige Präsenz des *Requirements-Engineers* während des ganzen Entwicklungsprozesses (vgl. Abschnitt 5.2) bewertet Heilig (2007) diese Dimension *(Beurteilungsdimension 2.2)* mit *2*, ohne dass die Werkzeugkette zum Einsatz kommt. Die Unterstützungskomponente bringt auch bei dieser Dimension keinen Erhöhung des Reifegrades mit sich. Trotz allem kann sie aber den *Requirements-Engineer* bei seiner Tätigkeit unterstützen, beispielsweise als Datenbank für den Nutzungskontext, d.h. die Ergebnisse dieser Analyse können strukturiert abgelegt werden und sind so für die anderen Mitarbeiter im Unternehmen frei zugänglich.

Usability-Engineer / Design

Der *Usability-Engineer* ist ständig am Entwicklungsprozess der Porsche AG beteiligt (vgl. Abschnitt 5.2). Mit dem Einsatz der Werkzeugkette beurteilt Heilig (2007) diese Dimension

(Beurteilungsdimension 2.3) mit dem *Reifestufe 2*. Über die Kommunikationswege des Wissensmanagementsystems, wie beispielsweise der Gruppendiskussionsfunktion, können Gestaltungsvorschläge unter den Entwicklern enger abgestimmt werden, ohne dass die verantwortlichen Personen am gleichen Standort sein müssen. Die Entwürfe werden einfach auf der Wissensplattform abgelegt und sind infolgedessen frei für die autorisierten Mitarbeiter zugänglich. Somit werden eventuelle Kommunikationsbarrieren zwischen der Entwicklern minimiert. Die Unterstützungskomponente hat aber keinen Einfluss auf eine Erhöhung der Reifestufe, da beispielsweise *Indikator 2.3.5* eine Fachverantwortung für die Usability-Entwürfe vorsieht.

Usability-Prüfer / Prüfung

Der *Usability-Engineer* führt die Prüfung der Produkte im HMI-Entwicklungsprozess selbst durch. Deshalb wird diese Dimension *(Beurteilungsdimension 2.4)* von Heilig (2007) mit *Reifestufe 1* klassifiziert.
Eine Aufteilung der Rollen des *Usability-Engineers* und des *Requirements-Engineers* würde eine höhere Einstufung des Reifegrades mit sich bringen. Mithilfe der Beratungskomponente erhöht sich die Qualität des Prototypen und des Produkts. So wird der *Usability-Prüfer* in dieser Phase bei der Bewertung der Gebrauchstauglichkeit entlastet, da das System schon frühzeitig eventuell auftretende Fehler erkennt. Dieser Aspekt der Unterstützung reicht jedoch nicht aus, um eine höhere Reifestufe zu erreichen, zumal das System keinen Einfluss auf die Rollenverteilung (*Usability-Engineers* und des *Requirements-Engineers*) nehmen kann.

Moderator / Moderation

Am HMI-Entwicklungsprozess ist ein interdisziplinäres Team von Entwicklern beteiligt. Bei einem wöchentlichen Treffen (Arbeitskreis GUI) werden in einer durch den Usability Engineer moderierten Runde Lösungsvorschläge diskutiert. Heilig (2007) bewertet diese Dimension *(Beurteilungsdimension 2.5)* mit der *Reifestufe 1*, da die Einbindung von späteren Nutzern nicht vorgesehen ist. In der Wissensplattform könnte das Protokoll der Sitzung transparent abgespeichert werden, um so zu einem späteren Zeitpunkt eventuelle Entscheidungen nachvollziehen zu können. Dies wurde auch deutlich von den Entwicklern bei der Befragung (vgl. Abschnitt 2.3) gewünscht. Zusätzlich könnten schon vor Beginn des Arbeitskreises GUI eine Agenda oder eventuelle Materialien wie PowerPoint-Folien in der Wissensplattform bereitgestellt werden. So könnten sich die Entwickler speziell auf das Treffen vorbereiten. Abschließend ist festzuhalten, dass die Unterstützungskomponente bei dieser Dimension den Reifegrades bewirkt, da keine späteren Benutzer am Arbeitskreis GUI teilnehmen.

Personenqualifizierung

Nach der Analyse von Heilig (2007) hat die Werkzeugkette keinen Einfluss auf die Dimension *(Beurteilungsdimension 3)* der Personenqualifizierung. In dieser Dimension kann mit dem Wissensmanagementsystem *Reifestufe 1* erzielt werden. Die Entwickler werden in den Bereichen *Analysemethodiken*, *Ableitung von Nutzeranforderungen* sowie *ergonomischer Normung* unterstützt. Die Komponente stellt den Entwicklern umfassendes Material, wie beispielswei-

se Fragebögen und Nutzeranforderungen aus früheren Projekten, zur Verfügung. Aus diesem Grund erhöht sich die Personalqualifizierung. Eine weitere Erhöhung des Reifegrades ist mit der Unterstützungskomponente nicht möglich, da das System nicht die Ansprüche der nötigen *Usability-Qualifizierung* von einzelnen Mitarbeitern messen kann.

Zeitpunkt des Einstiegs in das Projekt

Der *Usability-Engineer* ist während des ganzen HMI-Prozesses an der Entwicklung beteiligt (vgl. Abschnitt 5.2). Die in der Arbeit von Heilig (2007) beschriebene Werkzeugkette hilft dem *Usability-Engineer* zusätzlich bei der Erstellung von Prototypen. Somit können schon in sehr frühen Phasen spätere Nutzer miteinbezogen werden. Diese Dimension *(Beurteilungsdimension 4)* erreicht den Reifegrad von *3*. Die Unterstützungskomponente kann schon während der Vorbereitung des Projekts mit in die Entwicklung integriert werden. Die Erfahrungen *(Lessons Learned)* aus früheren Projekten können transparent abgespeichert werden.

Aufgabenanalyse

Im aktuellen HMI-Entwicklungsprozess ist zwar eine Aufgabenanalyse vorgesehen (vgl. Abschnitt 5.1), jedoch werden keine näheren Angaben gemacht, wie diese derzeit bei der Porsche AG abläuft. Dabei kommt eine methodische Entwicklung der Anforderungen wie beispielsweise *Use-Cases* zum Einsatz. Heilig (2007) bewertet nach seiner Evaluierung diese Dimension *(Beurteilungsdimension 5)* mit Reifestufe *0*. Die Werkzeugkette kann möglicherweise eingesetzt werden, um die Aufgaben zu modellieren und diese dann anhand von Prototypen zu prüfen. Diese Methode erhöht aber den Reifegrad nicht, da derzeit keine konkrete Vorgehensweise im Entwicklungsprozess etabliert ist. Die Wissensplattform kann wiederum dafür eingesetzt werden, die Analyseergebnisse aus dem Nutzungskontext transparent bereit zu stellen. Die Beratungskomponente unterstützt die Entwickler bei der Gestaltung von Prototypen. Aber allein anhand dieser Funktion kann der Reifegrad nicht erhöht werden.

Anforderungsentwicklung

In der Anforderungsentwicklung des HMI-Entwicklungsprozesses werden aktuelle Marktbeobachtungen durchgeführt (vgl. Abschnitt 5.1). Diese könnten in der Wissensdatenbank strukturiert nach Themenbereichen eingepflegt werden. Somit ist der *Indikator 6.1 (Ergebnisse aus Marktbeobachtungen werden berücksichtigt)* voll erfüllt. Heilig (2007) beschreibt, dass der aktuelle Prozess in dieser Dimension *(Beurteilungsdimension 6)* mit *Reifestufe 1* bewertet werden kann. Durch den Einsatz der Werkzeugkette können Usability Anforderungen anhand von Prototypen verifiziert werden. Demzufolge steigt die Qualität des Produktes, da mithilfe der Prototypen das frühe Einbeziehen der späteren Nutzer gewährleistet werden kann. Die Unterstützungskomponente hilft durch die Bereitstellung einer Datenbank, die Anforderungen strukturiert abzulegen. Zusätzlich werden die Entwickler bei der Gestaltung von ergonomischen Prototypen mit der integrierten Beratungskomponente in der Werkzeugkette unterstützt. Dies reicht aber nicht aus, um den Reifegrad des HMI-Entwicklungsprozesses zu erhöhen.

Usability-Prototyping

Durch den Einsatz der Werkzeugkette können schon in frühen Phasen des Entwicklungsprozesses erste Probandentests durchgeführt werden. Um jedoch den Reifegrad der Dimension *(Beurteilungsdimension 7)* von *1* zu erhöhen, müssten die zukünftigen Nutzer noch stärker miteingebunden werden (Heilig 2007). Die Prozessqualität kann durch den Einsatz der Unterstützungskomponente verbessert werden, indem die Entwickler bei der Gestaltung der Prototypen,durch die Bereitstellung ergonomischer Widgets, entlastet werden. Zusätzlich werden Usability relevante Normen in der Wissensdatenbank (vgl. Abschnitt 5.5) transparent in den einzelnen Projektphasen dargestellt. Jedoch reichen diese Maßnahmen nicht aus, um den Reifegrad und somit die Prozessqualität nach DATech (2006) zu erhöhen.

Prüfung

Derzeit basiert die Prüfung des Bediensystems auf der Grundlage des Porsche Styleguides (Porsche 2000). Dieser liegt aber nur in Papierform vor. Heilig (2007) bewertet diese Dimension *(Beurteilungsdimension 8)* mit *Reifestufe 1*. Mithilfe der Werkzeugkette kann beispielsweise einen Testfallgenerierung durchgeführt werden, dies berücksichtigt aber die Usability Normen nicht. Genau an diesem Punkt setzt die Beratungskomponente an: Mithilfe der Metriken, die in der Regelbasis abgelegt wurden, können schon bei der Gestaltung der Prototypen einschlägige Normen und Standards im Bereich des *Oberflächen-*, *Informations-* und *Interaktionsdesigns* berücksichtigt werden. So wird beispielsweise in der sehr frühen Phase der Gestaltung von Oberflächen der Abstand der Randpunkte nach der ISO 15008 überprüft (vgl. Abschnitt 5.4.2). Oder im Bereich des Interaktionsdesigns wird die Menütiefe im Meta CASE-Tool nach der Vorgabe von Normen (vgl. Abschnitt 5.4.2) kontrolliert. Allerdings kann die Reifestufe durch die Unterstützungskomponente nicht erhöht werden, dazu müssten wiederum repräsentative Nutzer bei der Prüfung beteiligt sein.

Dokumentation von Entwurfsentscheidungen

Der aktuelle HMI-Entwicklungsprozess in Verbindung mit der Werkzeugkette wird von Heilig (2007) in dieser Dimension *(Beurteilungsdimension 9)* mit *Reifestufe 2* bewertet, da derzeit schon Projektdokumentationen in Zusammenarbeit mit anderen Entwicklungsarbeiten zur Verfügung gestellt werden. Die Werkzeugkette dient hier als Versionsverwaltung für die modellierten Modelle. Durch den Einsatz der Wissensplattform kann der *Indikator 9.4* des Prüfhandbuches voll erfüllt werden. Dieser fordert neben einem Dokumentationssystem, in dem die Dokumente zum aktuellen Projekt aufbewahrt werden, zusätzlich ein System, das die Erfahrungen für das folgende Projekt bereitstellt. Die Erfahrungen enthalten beispielsweise nützliche Vorgehensweisen, Bewertungsergebnisse oder bewährte Lösungsvorschläge, die während der Produktentwicklung aufgetreten sind. Aus diesem Grund erhöht sich der Reifegrad durch die Einbindung der Wissensplattform auf *3*.

Einsatz und Anpassung von Methoden und Werkzeugen

Im aktuellen HMI-Entwicklungsprozess sind keine Usability-Engineering-Methoden vorgeschrieben. Dies wurde durch die Befragung (vgl. Kapitel 2) deutlich. Die Werkzeugkette

bringt einen erheblichen Vorteil, da die Entwickler an ein standardisiertes Werkzeug gebunden werden (Bock 2007). Das Problem ist jedoch, dass die Werkzeugkette nicht in allen Phasen (nur *Serienentwicklung*) des Entwicklungsprozesses eingesetzt werden kann. Eine erhebliche Verbesserung während des ganzen Entwicklungsprozesses erzielt jedoch die Unterstützungskomponente. Die in die Werkzeugkette integrierte Beratungskomponente sichert die konsistente Erscheinung der Oberfläche *(Indikator 10.2)*. Darüber hinaus existieren vorherige Projekterfahrungen *(Indikator 10.4)* der Entwickler auf der Wissensplattform. Ferner kann der Produkt-Styleguide während des Entwicklungsprozesse erweitert werden, wenn eventuelle Gestaltungsprobleme auftauchen, d.h. der Porsche Styleguide kann jederzeit von einer autorisierten Person normkonform erweitert werden *(Indikator 10.6)*. Somit kann mit der angedachten Unterstützungskomponente in dieser Dimension (Beurteilungsdimension 10) *Reifestufe 3* erreicht werden.

Einbettung in den Software-Entwicklungsprozess

Im jetzigen HMI-Entwicklungsprozess werden die Nutzungsanforderungen aus der Analyse des Nutzungskontextes berücksichtigt (Heilig 2007). Darüber hinaus werden die Entwurfsentscheidungen im Entwicklungsprozess mit Blick auf ihre Auswirkungen auf die Nutzungsqualität untersucht *(Indikator 11.3)*. Diese Dimension kann daher mit der *Reifestufe 2* bewertet werden. Nach Heilig (2007) hat die Werkzeugkette keinen Einfluss auf diesen Punkt. Die Nutzung der Unterstützungskomponente PoCKET würde bei dieser Dimension *(Beurteilungsdimension 11)* keinen Einfluss nehmen.

Benutzerdokumentation

Das Benutzerhandbuch der entwickelten Bediensysteme wird aus dem zuvor erarbeiteten Nutzungskonzept abgeleitet. Anhand von Grafiken und den gut ausformulierten Bedienschritten wird dem Benutzer eine hochwertige Dokumentation bereitgestellt (Porsche 2004), sodass die Dimension *(Beurteilungsdimension 12)* mit *Reifestufe 3* zu bewerten ist. Die Unterstützungskomponente erleichtert das Abgelegen der auftretenden Informationen während des Entwicklungsprozesses. Diese können auch jederzeit von jedem autorisierten Mitarbeiter ergänzt bzw. abgerufen werden. Außerdem werden die bisherigen Projekterfahrungen von der Wissensplattform bereitgestellt. Diese können zusätzlich in die didaktische Gestaltung des Trainingsmaterials einfließen *(Indikator 12.5)*. Mithilfe der Unterstützungsfunktion kann der Reifegrad, der ohnehin schon bei *3* liegt, nicht erhöht werden. Dennoch stellt die Komponente eine Erleichterung bei der täglichen Arbeit dar.

Personenunabhängigkeit

Der derzeitige HMI-Entwicklungsprozess wurde von Heilig (2007) mit Reifestufe *1* bewertet, da die Werkzeugkette keinen Einfluss auf diese Dimension *(Beurteilungsdimension 13)* hat. Der Usability-Engineer trägt letztlich die Verantwortung bei der Gestaltung von ergonomischen Benutzeroberflächen im HMI-Entwicklungsprozess. Deshalb ist nur der erste Indikator erfüllt. Die Komponente *PoCKET* unterstützt den *Indikator 13.3 (Ein Entwickler kann auf die Projekterfahrungen von Kollegen aufsetzten)* und den *Indikator 13.4 (Projekterfah-*

rungen werden während des Entwicklungsprozesses dokumentiert). Darüber hinaus können Arbeitsplatzbeschreibungen auf der Wissensplattform abgelegt werden, womit eine weitgehende Personenunabhängigkeit gesichert wird *(Indikator 13.2.)*. Dies kann beispielsweise der Fall sein, wenn ein Mitarbeiter erkrankt oder sogar das Unternehmen verlässt. Somit kann in diesem Punkt Reifestufe *3* erlangt werden.

Definition des Projektendes

Wie in Abschnitt 5.1 beschrieben, werden in den *Lessons Learned* die bisherigen Erfahrungen festgehalten, um diese dann anschließend in neue Projekte mit einfließen zu lassen. Darüber hinaus wird ab dem Zeitpunkt *Start-of-Produktion* (vgl. Abschnitt 5.1) zur *Serienbetreuung* ein *Change Control Board* eingeführt, um die eventuellen Qualitätsänderungen der Produkte aufzunehmen. Heilig (2007) bewertet den derzeitigen Prozess mit Reifestufe *2*, wobei die Werkzeugkette keinen Einfluss auf diese Dimension *(Beurteilungsdimension 14)* hat. Mithilfe der Wissensplattform kann der Reifegrad ebenfalls nicht erhöht werden. Die Entwickler können jedoch die Wissensplattform als Datenbank für die *Leasons Learned* und das *Change Control Board* benutzen.

Pflegeprozess

Der aktuelle Pflegeprozess sieht eine Unterstützung über die Phase der *Serienentwicklung* vor (vgl. Abschnitt 5.1). So sieht beispielsweise *Indikator 15.3* einen Kooperation zwischen Anwender und Hersteller vor, in der eventuell auftretende Fehler beseitigt werden. Dies wird über die Porsche Vertragshändler durchgeführt. Diese verständigen bei Mängeln das Entwicklungszentrum. Eventuell auftretende Fehler werden auf dem *Change Control Board* festgehalten. Somit kann der aktuelle HMI-Entwicklungsprozess mit *Reifestufe 3* bewertet werden, da alle Indikatoren berücksichtigt werden. Das Wissensmanagementsystem kann als Kommunikationsinstrument zwischen dem Entwicklungszentrum und den Porsche Vertragshändlern fungieren. Darüber hinaus kann das *Change Control Board* auf der Wissensplattform abgelegt werden.

5.6.2 Erkenntnisse aus der Bewertung

Mit der Unterstützungskomponente, welche im Rahmen dieser Arbeit entwickelt wurde, kann die Prozessqualität nach DATech 2006 in den Dimensionen *Dokumentation von Entwurfsentscheidungen (Beurteilungsdimension 9)*, *Einsatz und Anpassung von Methoden und Werkzeugen (Beurteilungsdimension 10)* und *Personenunabhängigkeit (Beurteilungsdimension 13)* erhöht werden, so dass *Reifegrad 3* erreicht werden kann. In der Beurteilungsdimension *Personenqualifizierung* kann die Reifestufe auf *1* erhöht werden. Abbildung 5.26 zeigt die Beurteilungsdimensionen mit den Reifegradstufen vor und nach dem Einsatz der Unterstützungskomponente *PoCKET*. Die Erhöhung der Reifestufen wurde gekennzeichnet. Auch mit der zusätzlichen Unterstützungskomponente kann der HMI-Entwicklungsprozess nicht mit einer höheren Reifestufe als in der Arbeit von Heilig (2007) bewertet werden. Die Reifestufe bleibt daher bei *0*, da nicht erfüllte Dimensionen nicht durch andere Kriterien des DATech-Prüfhandbuches ausgeglichen werden können, obwohl mit der Komponente

PoCKET in drei Beurteilungsdimensionen *Reifestufe 3* erreicht wurde (vgl. Abbildung 5.26), die zuvor zwischen *0* und *2* lagen.

Beurteilungs-dimension	Name	Reifestufe vorher	Reifestufe naher
1.	Usability Qualitätsziele	1	1
2.1	Benutzerbeteiligung	0	0
2.2	Requirements Engineer /Analyse	2	2
2.3	Usability Engineer /Design	2	2
2.4	Usability Prüfer /Prüfung	1	1
2.5	Moderator / Moderation	1	1
3.	Personenqualifizierung	k. A.	1
4.	Zeitpunkt des Einstiegsin das Projekt	3	3
5.	Aufgabenanalyse	0	0
6.	Anforderungsentwicklung	1	1
7.	Usability - Prototyping	1	1
8.	Prüfung	1	1
9.	Dokumentation von Entwurfsentscheidungen	2	3
10.	Einsatz und Anpassung von Methoden und Werkzeugen	0	3
11.	Einbettung in den Software Entwicklungsprozess	2	2
12.	Benutzerdokumentation	3	3
13.	Personenunabhängigkeit	1	3
14.	Definition des Projektendes	2	2
15.	Pflegeprozess	3	3

Abbildung 5.26: Vergleich der Beurteilungsdimensionen

Dennoch bringt die Unterstützungskomponente einen eindeutigen Mehrwert mit sich. So können die Usability-Qualitätsziele *(Beurteilungsdimension 1)* transparent in dem vorgesehenen Projektplan dargestellt werden. Eine frühere Nutzerintegration würde die Prozessqualität in der Beurteilungsdimension *Benutzerbeteiligung (2.1)* verbessern. Der *Requirements-Engineer* kann die Ergebnisse aus den Analysen (Nutzungskontext) auf dem Wissensmanagementsystem strukturierter ablegen. Die Entwickler werden schon sehr viel früher bei der Gestaltung der Bediensystemes unterstützt. Dies wirkt sich auf die Beurteilungsdimension des *Usability-Prüfers (Dimension 2.4)* aus. Der *Usability-Prüfer* braucht die grundlegenden Usability-Normen nicht mehr zu prüfen, da diese schon früh im Entwicklungsprozess von der Beratungskomponente berücksichtigt werden. Somit hat er mehr Zeit für die Prüfung anderer wichtiger Normungen während der Entwicklung. Auch der *Moderator* des Arbeitskreises GUI kann seine Unterlagen auf der Plattform bereitstellen. So können sich die anderen Entwickler auf das wöchentliche Treffen vorbereiten. Das spart Zeit für die Einarbeitung in das Thema während des Treffens. Zusätzlich haben die Entwickler die Möglichkeit, bei Unklarheiten eventuelle Fragen vorzubereiten. Somit würde der Arbeitskreis GUI qualitativ noch bessere Ergebnisse erzielen. Auch die *Beurteilungsdimensionen 7 Usability-Prototyping* und *8 Prüfung* werden von der Beratungskomponente unterstützt. So wird der Entwickler während der Erstellung der Prototypen umfassend in den Bereichen des *Oberflächen-*, *Informations-* und *Interaktionsdesigns* entlastet. Die Prüfung findet schon während der Modellierung, beispielsweise durch die Verwendung von Metriken (vgl. Abschnitt 5.4.1), statt.
Ferner wird in allen Beurteilungsdimensionen mit der Wissensplattform die Kommunikation

unter den Mitarbeitern gefördert. Nach Kuhlen (2000) bringt ein elektronisches Kommunikationsforum die Expertisen aus unterschiedlichen Disziplinen zusammen. Dadurch würde die Entwicklung der Bediensysteme positiv beeinflusst. Darüber hinaus bauen die Systeme die Kommunikationshürden zwischen den Hierarchien und Autoritäten ab (Kuhlen 2000, (Kuhlen 2002).
Weiter kann der Prozess durch das frühe Einbinden der späteren Nutzer verbessert werden. Die Unterstützungskomponente *PoCKET* hat aber keinen Einfluss auf die vorgegebenen Methoden in den einzelnen Prozessphasen der *Vorentwicklung*, *Serienentwicklung* und *Serienbetreuung*. Die Komponente kann hier nur helfen, indem sie die Entwickler bei ihren Tätigkeiten unterstützt.

Kapitel 6

Fazit und Ausblick

Das letzte Kapitel fasst Ergebnisse der Arbeit zusammen, zeigt Vorteile des entwickelten Konzeptes auf und denkt über dessen zukünftige Verwendung sowie seinen Erfolg nach.
Die Forschung im Bereich werkzeugbasierter Unterstützungsmöglichkeiten für die Gestaltung von Benutzeroberflächen steht noch in ihren Anfängen. Das darin vorhandene Potential muss noch ausgeschöpft werden, da es durch die wachsenden Anforderungen an die Entwickler durch Vorgaben wie Normen, Standard und Styleguide immer wichtiger wird. In diesem Punkt könnte eine Werkzeugunterstützung bei der Gestaltung von Benutzeroberflächen eine zunehmend wichtige Rolle einnehmen. Denn die angemessene Gestaltung von Benutzeroberflächen ist Grundlage für ein gebrauchstaugliches Produkt. Aus diesem Grund ist eine systematische Vorgehensweise bei der Gestaltung von Benutzeroberflächen, wie es im Rahmen dieser Arbeit durchgeführt wurde, unerlässlich. Darüber hinaus fordert der Entwicklungstrend den verstärkten Aufbau einer ergonomischen Kompetenz. So müssen im Rahmen des Usability-Engineerings während des ganzen Entwicklungsprozesses ergonomische Aspekte zum Einsatz kommen (Oed et al. 2001).

Der Vergleich der bestehenden Werkzeuge hat jedoch gezeigt, dass ein umfassender Ansatz, der von dieser Arbeit adressierten Problemstellung, noch nicht existiert. Und dies obwohl die grundlegenden Elemente des Usability Engineerings derzeit ausreichend dokumentiert sind. Nach Oed et al. (2001) liegt der Mangel in der fehlenden durchgängigen Unterstützung während der Bediensystemgestaltung.
Jedoch kristallisieren sich aus den betrachteten Methoden und Ansätzen einige Anknüpfungspunkte heraus. Im Rahmen dieser Arbeit wurden zahlreiche Formeln und Metriken analysiert und vorgestellt. Im Bereich des *Oberflächendesign* wurden Maße für Positionierung der Elemente, der Farbgebung sowie der Konsistenz untersucht. Diese Methoden stellen eine solide Basis für die Konstruktion von Unterstützungsmöglichkeiten bei der Erstellung von Benutzerschnittstellen dar. Das MetaCase-Tool bietet umfassende Unterstützung im Bereich des Interaktionsdesigns durch die Formalisierbarkeit der Transitionen und der Reports. Im Bereich des Informationsdesigns könnten über die Schnittstelle des CASE-Tools Anwendungen integriert werden, wie beispielsweise die Prüfung der Rechtschreibung. Für diese Komponente können ontologie-basierte Systeme eingesetzt werden, die semantische Beziehungen zwischen Begriffen in der Menüstruktur prüfen. So kann über einen Oberbe-

griff (Radio) der Unterbegriff (Senderwahl) semantisch zugeordnet werden.
Der in dieser Arbeit vorgeschlagene Ansatz, die Beratungskomponente in die Werkzeugkette zu integrieren, löst aber immer noch nicht das Problem zwischen dem GUI-Builder und den Language Workbeches. So haben die Symbole im MetaCASE-Tool derzeit keinen Bezug zu den im GUI-Builder gestalteten Oberflächenelementen. Diese müssen derzeit noch händisch nachgezogen werden. Der Ansatz mit den *Visual Studio DSL-Tools* muss weiter beobachtet werden. Dieser scheint den Bruch zwischen dem *Oberflächen-*, *Informations-* und *Interaktionsdesign* zu lösen. Im Konzept der vorliegenden Arbeit wurde mit *Microsoft Expression Blend* die nötige Schnittstelle geschaffen, um die Komponente enger zu verzahnen. Durch die Abkapselung des Wissensmanagementsystems kann die Anwendung völlig unabhängig von der Beratungskomponente betrieben werden.

Die vorgestellte Beratungskomponente und das Wissensmanagementsystem stehen in Wechselwirkung miteinander: Während verallgemeinerte Erfahrungen und Guidelines aus den aktuellen und vorangegangenen Projekten in das Wissensmanagementsystem einfließen, kann umgekehrt die Einhaltung bestimmter Guidelines und Standards durch die Beratungskomponente in die Entwicklung aufgenommen werden. Der Einsatz der beiden Komponenten stellt dadurch einen wichtigen Baustein eines integrierten Qualitätssicherungskonzeptes für die Entwicklung dar (Conrad et al. 2006). In Zukunft stellt sich die Frage, inwiefern die Automobilindustrie von den vorgestellten Konzepten profitieren kann. Folgende Vorteile sind möglich: Eine sehr hohe Transparenz bei der Gestaltung von Bediensystemen für alle am HMI-Entwicklungsprozess beteiligten Personen. Durch die hohe Transparenz im Prozess wird zusätzlich die zielführende Kommunikation und Erschließung von Synergiepotenzialen zwischen den Disziplinen gefördert. Das Festlegen von Entwicklungsstandards in sehr frühen Phasen würde ebenfalls einen Vorteil bei der Produktqualität mit sich bringen. Darüber hinaus sollten bei der Entwicklung die gesammelten Ergebnisse, Erfahrungen und Erkenntnisse festgehalten werden. So wird verhindert, dass bei Folgeprojekten nicht ständig das *Rad neu erfunden* werden muss. Diese Aspekte führen zu einer ständigen Verbesserung des Entwicklungsprozesses. Die oben genannten Punkte decken sich mit den Anforderungen der Entwickler, welche im 2. Kapitel dargelegt wurden.

Die Entwicklungsprozesse haben, wie zuvor beschrieben, einen außerordentlichen Einfluss auf die Verbesserung der Produkte. Die in dieser Arbeit gezeigten Ansätze können für die Verbesserung des HMI-Entwicklungsprozesses verwendet werden. Essentiel ist aber die Akzeptanz des HMI-Entwicklungsprozesses bei allen beteiligten Personen. So sollte bei der Einführung neuer Werkzeuge in den Entwicklungsprozess bei der Vorbereitung neben der Automatisierung von Prozessteilen auch auf die ausreichende methodische Schulung geachtet werden. Dabei müssen die am HMI-Entwicklungsprozess beteiligten Personen im Vordergrund stehen. Nach Maier (2006) ist für den erfolgreichen Prozesseinsatz wichtig, dass die Prozessbeteiligten auch die nötige Zeit für ihre Arbeit eingeräumt bekommen, da die Fertigkeiten und der Wille eine gute Arbeit abzuliefern, nahezu immer bei den Mitarbeitern vorhanden sind.

So lässt sich abschließend festhalten, dass es als sehr sinnvoll erscheint, Beratungs- und Wissensmanagementkomponenten während der Entwicklung interaktiver Systeme in die Prozesse zu integrieren.

Abbildungsverzeichnis

1.1 Fahrerinformationssystem PCM 2.1 der Porsche AG 5

2.1 Am Entwicklungsprozess beteiligte Personen 10
2.2 Auswertung der Kategorien mit priorisierte Rangfolge in Anlehnung an Bock (2007) 12
2.3 Bisherige Vorgehensweise 13
2.4 Häufigkeitsverteilung Spezifikationswerkzeuge 14
2.5 Rollen und Werkzeuge im HMI-Entwicklungsprozess 14
2.6 Aufbau der Werkzeugkette (Memmel et al. 2007) 17
2.7 Klassifizierung der vorgestellten Forschungsprojekte (Heilig 2007) 18

3.1 Usage-Centered-Design Constantine und Lockwood (1999) 23
3.2 Useware-Entwicklungsprozess nach Zülke et al. (2004) 24
3.3 Schritte des DIN EN ISO 13407 Gestaltungsprozess für interaktive Systeme . 26
3.4 Grad der Automatisierung in Anlehnung an Beirekdar (2004) 29
3.5 Klassifikation von Hilfesystemen 31
3.6 Auskunfts- und Beratungssysteme in Anlehnung an Balzert (1998) 33
3.7 Explizites Wissen vs. Implizites Wissen 36
3.8 Beispiele für Wissensträger 37
3.9 Stufen Modell nach Bohn (1994) 37
3.10 Praxisaspekte des Wissensmanagements (Studer et al. 2001) 38
3.11 Prozessmodell nach Probst (1998) 39
3.12 Barrieren beim Wissenstransfer (Leitner und Tuppinger 2004) 41

4.1 Softwareentwicklung mit MUSE (Gorny 1997) 45
4.2 Komponenten von ProUSE (Metzker 2005) 47
4.3 Werkzeuge zur Gestaltung von ergonomischen Benutzeroberfächen 49
4.4 Konzept von KWARESMI (Beirekdar 2004) 52
4.5 Werkzeuge zur automatischen Bewertung von Benutzeroberflächen 54
4.6 IDA Entwicklungsumgebung (Reiterer 1995) 55
4.7 Vorgehensmodell zur Entwicklung von Assistenzsystemen Wandke et al. (2002) 56
4.8 Der GUIDE Prozess (Henninger 2000) 57
4.9 Beratungswerkzeuge zur ergonomischen Gestaltung von Benutzeroberflächen . 58

5.1 HMI-Entwicklungsprozess der Porsche AG (Bock 2007) 62

5.2 Projektphasen und Rollen 65
5.3 Bedienelemente PCM 2.1 70
5.4 Aufteilung der Komponenten 71
5.5 Überprüfung der Oberfläche (Beirekdar et al. 2002) 73
5.6 Berechnung der Kanten 75
5.7 Spektraler Farbabstand (Hamacher 2006) 77
5.8 Klassifizierung der Beratung 81
5.9 Mockup-XAML GUI-Builder mit Geometry-Control und Appearance Panel . 83
5.10 Mockup Color-Control Panel 83
5.11 Mockup Font-Control Panel 86
5.12 Bibliothek Projekt PCM 2.1 86
5.13 Mockup-Wortergänzung 87
5.14 Mockup Längenprüfung der Beschriftung 88
5.15 Transitionen in MetaEdit+ 89
5.16 Fehlermeldung bei Benutzung von falschen Bedienelementen 90
5.17 Erlaubte Transition der Hauptbedientaste 91
5.18 Konzept Architektur für die Beratungskomponente 92
5.19 Vereinfachte Struktur einer Wissensplattform 94
5.20 BKM-Toolarchitektur (Bach und Österle 1999) 95
5.21 Aufbau des Porsche Wissensmanagementsystems 96
5.22 Die Porsche AG Wissenspyramide 98
5.23 Mockup Textmanagement-Tool 100
5.24 Benutzeroberfläche Wissensmanagementsystem 103
5.25 Einpflegen Styleguide in die Wissensplattform 105
5.26 Vergleich der Beurteilungsdimensionen 113

C.1 Beispiel PCM Bildschirm 139

Abkürzungsverzeichnis

AK	Arbeitskreis
BHA	Benutzerhäufigkeitsanalyse
BKM	Business Knowledge Management
CAN	Controller Area Network
ca.	circa
DDS	Dreh-Drück-Steller
d. h.	das heißt
DSL	domänenspezifische Modellierungssprache
DIN	Deutsche Industrie Norm
EEI	Elektrik/Elektronik-Infotainment
EEV	Elektrik/Elektronik-Vorentwicklung
GUI	Graphical User Interface
HMI	Human-Machine Interface
HTML	Hypertext Markup Language
ISO	International Standard Organization
MMI	Mensch-Maschine-Interaktion
MOST	Media Oriented Systems Transport
MS DSL Tools	Microsoft Domain Specific Language Tools
PCM	Porsche Communication Management
UI	User-Interface
UIMS	User Interface Management System
vgl.	vergleiche
WPF	Windows Presentation Foundation
XAML	eXtensible Application Markup Language
XML	Extensible Markup Language
z.B.	Zum Beispiel

Literaturverzeichnis

[Aaronson und Carroll 1986] *'Aaronson A., Carroll J.M. : The Answer is in the Question: A Protocol Study of Intelligent Help, IBM Research Report, IBM Watson Research Center, New York, 1986'*

[Bach et al. 1999] *'Bach V., Vogler P., Österle H.: Business Knowledge Management: Praxiserfahrungen mit intranetbasierten Lösungen, Heidelberg, 1999'*

[Balzert 1998] *'Balzert H.: Lehrbuch der Software Technik - Software-Management Software-Qualitätssicherung Unternehmensmodellierung, Heidelberg, 1998'*

[Balzert 1994] *'Balzert H.: Das JANUS-System: Automatisierte, wissensbasierte Generierung von Mensch-Computer-Schnittstellen, Informatik-Forschung und Entwicklung 9, Heidelberg, 1994, Seiten 22-35'*

[Bauer 1988] *'Bauer J.: Konzepte und Prototypen interaktiver Hilfesysteme, Fakultät Mathematik und Informatik, Universität Stuttgart, 1988'*

[Bauer und Schwab 1988] *'Bauer J., Schwab T.: Anforderungen an Hilfesysteme, in: H. Balzert (Hrsg.): Einführung in die Software-Ergonomie, New York, 1988'*

[Beirekdar et al. 2002] *'Beirekdar A., Vanderdonckt J., Noirhomme M.: KWARESMI - Knowledge-based Web Automated Evaluation with Reconfigurable guidelines optimization, Paris, 2002'*

[Beirekdar 2004] *'Beirekdar A.: Methodology for Automating Guidelines Review of Web Sites, Paris, 2004'*

[Beyer und Holtzblatt 1998] *'Beyer H., Holtzblatt K.: Contextual Design. Defining Customer-Centered Systems, San Francisco, 1998'*

[Blume et al. 2005] *'Blume M., Stokar D., Seewald F.: Usability-Evaluation: Egal wer's macht? Ein schweizer Fallbeispiel. i-com, Zürich, 2005, Seiten 69-71'*

[Bock 2007] *'Bock C.: Einsatz formaler Spezifikationen im Entwicklungsprozess von Mensch-Maschine-Schnittstellen, Fachbereich Maschinenwesen und Verfahrenstechnik, Technischen Universität Kaiserslautern, 2007'*

[Bohn 1994] *'Bohn R.: Measuring and Managing Technological Knowledge, in: SMR, Vol. 36, Nr. 1 (Fall-Issue), 1994, Seite 61-73'*

[Borchers 2001] *'Borchers J., Pattern A.: Approach to Interaction Design Department of Computer Science, Darmstadt University of Technology, 2001'*

[Budde und Züllighoven 1990] *'Budde R., Züllighoven H.: Software-Werkzeuge in einer Programmierwerkstatt, Bericht der GMD, Nr. 182, Oldenbourg, 1990'*

[Bügel 2004] *'Bügel U.: Projekt WiMan Globaler Leitfaden Wissensmanagement, Fraunhofer IITB, 2004'*

[Burmester et al. 2005] *'Burmester M., Machate J., Sandweg N.: Integration benutzerzentrierter Methoden in die Software-Entwicklung, Stuttgart I-Com, 2005.'*

[Capurro 1998] *'Capurro R.: Wissensmanagement in Theorie und Praxis, Forschungsbereicht, 1998'*

[Cherri et al. 2004] *'Cherri C., Nodari E., Toffetti A.: Adaptive integrated driver-vehicle interface (AIDE), Review of existing Tools and Methods, 2004 '*

[Conrad et al. 2006] *'Conrad M., Dörr H., Fey I., Pohlheim H., StürmerI.: Guidelines und Reviews in der Modell-basierten Entwicklung von Steuergeräte-Software, 2006 '*

[Constantine und Lockwood 1999] *'Constantine L., Lockwood L. A.D.: Software for use: a practical guide to the models and methods of usage-centered design, 1999 '*

[Davenport und Prusak 1998] *'Davenport T., Prusak, L.: Working Knowledge. How Organizations manage what they know, Boston, 1998 '*

[DATech 2006] *'DATech: DATech-Prüfhandbuch - Usability-Engineering-Prozess. Leitfaden für die Evaluierung des Usability-Engineering-Prozesses bei der Herstellung und Pflege von interaktiven Systemen auf der Grundlage von DIN EN ISO 13407, Version 1.4, DATech - Deutsche Akkreditierungsstelle Technik GmbH, 2006'*

[Delphi-Group 1998] *'Delphi-Group: Wissen im Unternehmen , 1998 http://www.ifem.org (Letzter Zugriff: 16.07.2007)'*

[De Jong 2004] *'De Jong R.: Detailtypografie -Nachschlagewerk für alle Fragen zu Schrift und Satz, 2004'*

[DIN EN ISO 13407 1999] *'DIN EN ISO 13407: Human-centred design processes for interactive systems. Ed. 1, Current stage 90.93, TC 159/SC 4, 1999'*

[DIN EN ISO 9241 1997] *'DIN EN ISO 9241 : Ergonomic requirements for office work with visual display terminals (VDTs). Ed. 2, Current stage 90.93, TC 159/SC 4, 1997'*

[Duffy et al. 1992] *'Duffy T., Palmer M., Mehlenbacher B.: On Line Help, Design and Evalution, New Jersey, 1992'*

[Drucker 1992] *'Drucker P.: Die Zukunft managen, Düsseldorf, 1992'*

[Dunckel et al. 1993] *'Dunckel H., Volpert, Walter K., Pleiss C. ,Hennes K.: Konstrastive Aufgabenanalyse im Büro, Der KABA-Leitfaden. Grundlagen und Manual, Stuttgart, 1993'*

[Eberle 2003] *'Eberle A.,: Barrieren und Anreizsysteme im Wissensmanagement und der Software-Wiederverwendung, Fraunhofer Institut, Experimentelles Software Engineering, 2003'*

[European Commission 1999] *'European Commission: Safe and Efficient In-Vehicle Information and Communication Systems: A European Statement of Principles onHumen-Machine-Interfaces, 1999'*

[Farenc 1997] *'Farenc C., ErgoVal: une M´ethode de Structuration des R'egles Ergonomiques Permettantl'´Evaluation Automatique d'Interfaces Graphiques, Universit´e Toulouse, 1997'*

[Forssman und de Jong 2004] *'Forssman F., de Jong R.: Detailtypografie. Nachschlagewerk für alle Fragen zu Schrift und Satz, 2004'*

[Fraunhofer Wissensmanagement Community 2005] *'Fraunhofer Wissensmanagement Community: Wissen und Inforamtion, 2005'*

[Freise 2003] *'Freise A.: Der Nutzen gut bedienbarer Produkte. Siemens - Pictures of the Future, 2:65. Siemens AG, München, 2003'*

[Frieling und Reuther 1993] *'Frieling E., Reuther U.: Das lernende Unternehmen. Reihe: Studien der betrieblichen Weiterbildungsforschung, Bochum, 1993'*

[Geis 2005] *'Geis T.: Das DATech-Prüfhandbuch Usability-Engineering-Prozess. Ein Verfahren zur Überprüfung der Normkonformität mit DIN EN ISO 13407, Fit für Usability, Fraunhofer-Institut, 2005'*

[Gomma et al. 2005] *'Gomma M., Salah A., Rahman S.: Towards A Better Model Based User Interface Development Environment: A Comprehensive Survey. In: Proceedings of the 38th Annual Midwest Instruction and Computing Symposium, 2005'*

[Gorny und Nordby 1995] *'Gorny P., Nordby K.: Human-Computer Interaction, Proc. Interact '95. IFIP Conf. on Human Computer Interaction. Norway, 1995 '*

[Gorny 1997] *'Gorny P.: Kontextbezogener Entwurf von Benutzungsoberflächen mit MUSE-II. Tutorial im Rahmen der Fachtagung Software-Ergonomie, Tutorial im Rahmen der Fachtagung Software-Ergonomie, Dresden, 1997'*

[Gould und Lewis 1985] *'Gould J., Lewis C.: Designing for Usability: Key Principles and What Designers Think. Communications of the ACM, 28(3), 1985, Seiten 300-311.'*

[Grammenos et al. 2000] *'Grammenos D., Akouminanakis D., Stephanidis C.: Integrated support for working with guidelines: the Sherlock guideline management system, Science and technilogy Park of Crete, in Interacting with Computers, 2000'*

[Green 1986] *'Green M.: A survey of three dialogue models. ACM Transactions on Graphics 5 ,Nr. 3, 1986, Seiten 244-275'*

[Hamacher 2006] *'Hamacher N.: Automatische kriterienorientierte Bewertung der Gebrauchstauglichkeit interaktiver Systeme, Elektrotechnik und Informationstechnik der Rheinisch-Westf¨alischen Technischen Hochschule Aachen, 2006'*

[Heilig 2007] *'Heilig H.: Integration modellgetriebener Methoden in eine Usability-Entwicklung, Praktische Erprobung und Einbettung in den HMI-Entwicklungsprozess der Dr. h.c. F. Porsche AG, 2007'*

[Henninger 2000] *'Henninger S.: A Methodology and Tolls for Appling Context-Specific Usability Guidelines to Interface Design, Departmant of Comuter Sciencen & Engineering, University of Nebraska-Licoln, 2000'*

[Henninger 2003] *'Henninger S.: Tool Support for Experience-Based Software Development Methodologies Appears in: Advances In Computers, University of Nebraska-Lincoln, 2003, Seiten 29-82'*

[Hurtienne und Prümer 2006] *'Hurtienne J., Prümer J.,: Vom Zauberer zum Partner-Usability Beratung im Spiegel organisationaler Reife, Consulting Research-Unternehmensberatung aus wissenschaftlicher Perspektive, Wiesbaden: Deutscher Universitätsverlag, 2006, Seiten 309-327'*

[Hüttner et al. 1995] *'Hüttner J., Wandke, H., Rätz A.: Benutzerfreundliche Software, Psychologisches Wissen für die ergonomische Schnittstellengestaltung. Bernd-Michael Paschke Verlag, Berlin, 1995.'*

[ISO 15005] *'ISO 15005: Road vehicles-Ergonomic aspects of transport information and control systems-Dialogue management principles and compliance procedures, 2002'*

[ISO 15008] *'ISO 15008: Road vehicles-Ergonomic aspects of transport information and control systems- Specifications and compliance procedures for in-vehicle visual presentation, 2003'*

[Ivory 2001] *'Ivory M. Y.: An Empirical Foundation for Automated Web Interface Evaluation, Computer Science Division, UC Berkeley, 2001'*

[Ivory und Hearst 2001] *'Ivory M. Y, Hearst M.: The State of the Art in Automating Usability Evaluation of User Interfaces. ACM Computing Surveys, 33(4), 2001, Seiten 470-516.'*

[Ivory et al. 2001] *'Ivory M. Y., Hearst M., Rashmi S.: Empirically Validated Web Page Design Metrics. In: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press, New York, Seiten 53-60'*

[Jansen et al. 2001] *'Jansen C., Blessing D., Bach V., Österle H.: Knowledge Portals for Business Engineering-A Reference Architecture, St. Gallen, 2001'*

[Janssen et al. 1993] *'Janssen J., Weisbecker A., Ziegler J.: Generierung graphischer Benutzungsschnittstellen aus Datenmodellen und Dialognetz-Spezifikationen, Stuttgart 1993'*

[Johnson 2000] *'Johnson, J.: GUI Bloopers: Don'ts and Do's for Software Developers and Web Designers, Oxford, 2000'*

[Kaiser 1998] *'Kaiser, T.: Grundlagen des Kontextmanagements. Arbeitsbericht HSG/IM SG/CC I-Net, Institut für Wirtschaftsinformatik, Universität St. Gallen, St.Gallen, 1998'*

[Kingsbury 2003] *'Kingsbury N.:D esign of q-shift complex wavelets for image processing using frequency domain energy minimization, IEEE Int. Conf. Image Processing, Barcelona, 2003, Seiten 1013-1016'*

[Kuhlen 2000] *'Kuhlen R.: Wissensmanagement über elektronische Kommunikationsforen, Universität Konstanz, 2000'*

[Kuhlen 2002] *'Kuhlen R.: Wissensmanagement über elektronische Kommunikationsforen, Erscheint in Proceedings SEL-ALCATEL-Summer school Berlin, 2002'*

[Labinsky 1996] *'Labinsky C.: Zur Methodik der Neuentwicklung von Benutzungsoberflächen: Beiträge für einen Leitfaden zur Method for user Interface Engineering (MUSE), Universität Oldenburg, 1996'*

[Lehner 2006] *'Lehner F.: Wissensmanagement, München, 2006, Seiten 253-272'*

[Leitner und Tuppinger 2004] *'Leitner W., Tuppinger J.: Wissenstransfer bei der virtuellen Teamarbeit, Prozesswissen als Erfolgsfaktor - Effiziente Kombination von Prozessmanagement und Wissensmanagement, 2004'*

[Limbourg und Vanderdonckt 2001] *'Limbourg Q., Vanderdonckt J.: Completing Human Factor Guidelines by Interactive Examples, Tools for Working with Guidelines, London, 2001, Seiten 357-373.'*

[Löwgren und Nordqvist 1992] *'Lowgren J., Nordqvist T.: Knowledge based evaluation as design support für gaphical user interface, Dept. of Computer and Info, Science Linkoping University Linkoping, Sweden, 1992'*

[Mahajan und Sheiderman 1997] *'Mahajan R., Sheiderman B.: Visual and Textual Consistency Checking Tools for Graphical User Interfaces. IEEE Transactions on Software Engineering, 1997, Seiten 722-735'*

[Mayhew 1999] *'Mayhew D.: The usability engineering lifecycle: a practitioner's handbook for user interface design. San Francisco, 1999'*

[Meier 2006] *'Meier E.: Angemessene Entwicklungsprozesse, 1999'*

[Melody et al. 2000] *'Melody Y. Ivory M., Hearst M.: The State of the Art in Automated Usability Evaluation of User Interfaces, Computer Science Division, University of California, 2000'*

[Memmel et al. 2007] *'Memmel T., Bock, C., Reiterer H.: Model-driven prototyping for corporate software specification. Arbeitsgruppe Mensch-Computer Interaktion, Universität Konstanz, Dr. Ing. h.c. F. Porsche AG, Germany, 2007'*

[Metzker 2005] *'Metzker E.: Adoption Centric Usability Enginnering Systematic Deployment, Evaluation and Improvement of Usability Engineering Methods in the Software Engineering Lifecycle, 2005'*

[Metzker und Reiterer 2002a] *'Metzker E., Reiterer H.: Use and Reuse of HCI Konwledge in the Software Development Lifecycle, Exsisting Approaches and what Developers Think, DaimlerChrysler Reascher Center Ulm, Universität Konstanz, 2002'*

[Metzker und Reiterer 2002b] *'Metzker E., Reiterer H.: Evidence-Based Usability Engineering, DaimlerChrysler Reascher Center Ulm, Universität Konstanz, 2002'*

[Moussa et al. 2000] *'Moussa F., Kolski C., Riahi M.: A model based approach to semi-automated user interface generation for process control interactive applications, Interacting with Computers, Volume 12, Issue 3, Tunisia, 2000, Seiten 245-279'*

[Ngo et al. 2000] *'Ngo D., Samsudin A., Abdullah R., Rosni: Aesthetic measures for assessing graphic screens. In: Journal of Information Science and Engineering, Malaysia, 2000, Seiten 97-116'*

[Nonaka und Takeuchi 1997] *'Nonaka I., Takeuchi H.: Die Organisation des Wissens. Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen, Frankfurt, 1997'*

[Normen 1988] *'Normen D.: The Psychology of Everyday Things, New York, 1998'*

[Oed et al. 2001] *'Oed R., Becker A., Wetzstein E.: Welche Unterstützung wünschen Softwareentwickler beim Entwurf von Bedienoberflächen?, DaimlerChrysler Forschung Ulm, Humboldt Universität Berlin, 2001'*

[Oppermann und Reiterer 1997] *'Oppermann R., Reiterer H.: Software Evaluation using the 9241 Evaluator. Behaviour & Information Technology, 1997, Seiten 232 - 245'*

[Oppermann et al. 1992] *Oppermann R., Murchner B., Reiterer H., Koch M.: Softwareergonomische Evaluation: Der Leitfaden EVADIS II, 1992*

[Porsche 2000] *'Porsche: HMI-Entwicklung - Vorgehen und Methoden, Porsche internes Dokument, 2000'*

[Porsche 2004] *'Porsche: Benutzerhandbuch PCM 2.1 (deutsch), Porsche internes Dokument, 2004'*

[Puppe 2003] *'Puppe F.: Knowledge Engineering: Handbuch der künstlichen Intelligenz, München, 2003'*

[Probst et al. 1998] *'Probst G., Raub S., Romhardt K.: Wissen managen. Wie Unternehmen ihre wertvollste Ressource optimal nutzen Frankfurt/Main, Frankfurter Allgemeine, 1998'*

[Probst et al. 2003] *'Probst G., Raub S., Romhardt K.: Wie Unternehmen ihre wertvollste Ressource optimal nutzen, Heidelberg, 2003'*

[Rauch und Totzke 2004] *'Rauch I., Totzke N.: Kompetenzerwerb für Fahrerinformationssysteme: Bedeutung von Bedienkontext und Menüstruktur / Interdiziplinäres Zentrum für Verkehrswissenschaften, Universität Würzburg, 2004'*

[Reed et al. 1999] *'Reed P., Holdaway K., Isensee S., Buie E., Fox J., Williams J., Lund A.: User interface guidelines and standards: progress, issues, and prospects, Lucent Technologies, USA, 1999'*

[Reiterer 1995] *'Reiterer H.: Die IDA-Entwicklungsumgebung: Einsatz von objektorientierten, multimedialen und wissensbasierten Unterstützungswerkzeugen zur ergonomischen Gestaltung von Benutzungsoberflächen, Informatik Forschung und Entwicklung, München, 1995, Seiten 180-196'*

[Reiterer 2000] *'Reiterer H.: Tools forWorking with Guidelines in Different Interface Design Approaches. In: Vanderdonckt, Farenc, Christellem : Tools for Working with Guidelines, Proc. of the Int. Workshop on Tools forWorking with Guidelines,France , October, 2000, Seite 225-236'*

[Robinson 1993] *'Robinson M.: Design for unanticipated use, Proceedings of the Third European Conference on Computer-Supported Cooperative Work, 13.-17. Italy, 1993'*

[Sauter 1999] *'Sauter W.: Methodik des vernetzten Denkens. In: Wissenschaftsmanagement 4, 1999, Seiten 25-30'*

[Schwarz 2005] *'Schwarz T.: Analyse der Benutzungshäufigkeit von Funktionen bei Fahrerinformationssystemen, Nicht freigegebene Diplomarbeit, Dr. Ing. h.c. F. Porsche AG und HS Ravensburg-Weingarten, 2005'*

[Scapin et al. 2001] *'Scapin D., Leulier J., Vanderdonckt C., Bastien C., Farenc C., Palanque P., Bastide R.: T.: A Framework for Organizing Web Usability Guidelines, Proceedings of 6th Conference on Human Factors and the Web, 2001'*

[Schmid 1999] *'Schmid R.: Wissensmanagement für den Innovationsprozeß, 1999'*

[Schulz 2001] *'Schulz H.: Ist ein Wissensmanager erfolgreich, macht er sich überflüssig. In: Frankfurter Allgemeine Zeitung, 29. Januar 2001, Seiten 1-2'*

[Schwaninger 2000] *'Schwaninger M.: Implizites Wissen und Managementlehre: Organisationskybernetische Sicht, Universität Ulm, 2000'*

[Sears 1995] *'Sears A.: AIDE: A step toward metric-based interface development tools. In: Proceedings of the 8th ACM Symposium on User Interface Software and Technology(UIST). Pittsburgh,ACM, 1995, Seiten 101-110'*

[Shneiderman 1998] *'Shneiderman B.: Designing the user interface: strategies for effective humancomputer- interaction, 3. Addison-Wesley, 1998'*

[Shneiderman Plaisant 2005] *'Shneiderman B., Plaisant C.: Designing the User Interface: Strategies for effective Human-Computer Interaction, 2005'*

[Smith und Mosier 1986] *'Smith S., Mosier J.: Guidelines for Designing User Interface Software, Mitre Corporation, Bedford, 1986'*

[Suchman 1987] *'Suchman L.A.: Plans and situated actions, The problem of human-machine comunication, Cambridge University Press, Cambridge, 1987'*

[Souza und Bevan 1990] *'Souza F., Bevan N.: The use of guidelines in menu interface design: evaluation of a draft standard. Proceedings of Interact, Human-Computer Interaction, Cambridge, 1990, Seiten 435-440'*

[Studer et a. 2001] *'Studer R., Schnurr H., Nierlich A.: Semantisches Knowledge Retrievel, Ontoprise Whitepaper, Series, 2001'*

[Thevenin 1999] *'Thevenin D.: Adaptation and Plasticity of User Interfaces, France, 1999'*

[Vanderdonckt 1999] *'Vanderdonckt J.: Development milestones towards a tool for working with guidelines, Interacting wirth Computers 12, Univertität Louvain, Belgien, 1999 Seiten 81-118'*

[Von der Beck et al. 2001] *'Von der Beck M., Braun, P., Rappl, M., Schröder, C.: Modellbasierte Softwareentwicklung für automobilspezifische Steuergeräte. BMW AG, München Technische Universität, Telelogics Deutschland GmbH, Bielefeld, 2001'*

[Wandkte und Nitschke 2002] *'Wandke H., Nitschke J.: Das Vorgehensmodell GUIDEAS: GUIdance for DEveloping Assistance, Humboldt-Universität Berlin, 2002'*

[Wandkte et al. 1999] *'Wandke H., Dobrowsky A., Hüttner J.: Anforderungsanalyse zur Einführung eines Unterstützungssystems bei Softwarentwicklern, Berichte des German Chapter of ACM, Stuttgart, 1999'*

[Wegner 2004] *'Wegner G.: Das elektronische Lastenheft als Mittel zur Kostenreduktion bei der Entwicklung der Mensch-Maschine-Schnittstelle von Infotainmentsystemen im Fahrzeug, IAV, Berlin, 2004'*

[Woletz 2006] *'Woletz N.: Evaluation eines User-Centred Design-Prozessassessments, Empirische Untersuchung der Qualität und Gebrauchstauglichkeit im praktischen Einsatz, Dissertation, Universität Paderborn, 2006'*

[Zaphiris et al. 2002] *'Zaphiris P., Shneiderman B., Norman K.: N.: Expandable Indexes vs. Sequential Menus for Searching Hierarchies on the World Wide Web. Behaviour and Information Technology, Virginia, 2002, Seiten 201-207.'*

[Zülke 1999] *'Zülke D., Wahl M.: Hardware, Software -Useware, Elektronik, Nr. 23, Kaiserslautern, 1999, Seiten 54-62'*

[Zülke 2004] *'Zülke D.: Useware-Engineering für technische Systeme, Berlin, 2004'*

Anhang A

Fragebogen

Konzeption einer rechnerbasierten Toolunterstützung für den HMI-Entwicklungsprozess

Motivation der Umfrage - Im Rahmen dieser Umfrage sollen die aktuell verwendeten Methoden und Werkzeuge im HMI-Entwicklungsprozess analysiert werden, um eine rechnerbasierte Toolunterstützung zu entwickeln bzw. zu evaluieren. Derzeit arbeiten Sie wahrscheinlich mit Excel, Word oder Powerpoint.

Darum bitten wir um Ihre Unterstützung!

Position: _____

A. Aufgabenbereich

1. Beschreiben Sie ihren Aufgabenbereich im Umfeld des HMI-Entwicklungsprozesses.

_____

_____

_____

_____

_____

_____

_____

2. Wenn die Entwicklung von Mensch-Maschine-Schnittstellen für Fahrerinformationssysteme (z.B. PCM) in 3 Arbeitspakete (Design, Inhalt und Bedienabläufe) eingeteilt werden würde, in welches Paket würden Sie Ihre Arbeit einordnen?

☐ Design (Blendendesign, Screendesign, Layout, Anordnung, Farben, Schriften …)

☐ Inhalt (Menüpunkte, Wording)

☐ Bedienabläufe (Bedienkonzept, Menüstruktur…)

B. Beurteilung der Kriterien

Bitte bewerten Sie folgende Anforderungen, an eine DV-Werkzeugunterstützung für den HMI-Entwicklungsprozess . Wenn Sie zu einem Punkt keine Aussage treffen können, dann überspringen Sie diesen Punkt.

Anforderungen	Beschreibung	weniger wichtig	wichtig	sehr wichtig
Funktionsumfang				
1. Anpassbarkeit	Anpassbar für verschiedene Projekte (PCM, CDR, Kombiinstrument, Klima, Bedienhörer ...)	☐	☐	☐
2. Modellierung Bedienabläufe	Graphische Modellierung Bedien-konzept, Struktur, Vernetzung Menüpunkte+Screens	☐	☐	☐
3. Vertraute Notation	Symbole, Begriffe aus Aufgaben-bereiche (z.B. DDS, Hauptbedien-taste, Telefonsymbol ...)	☐	☐	☐
4. Look & Feel	Intuitive, schön gestaltete Oberflächen	☐	☐	☐
5. Wiederverwendbarkeit	Vorlagen, Bildschirme duplizieren, vordefinierte Bedienabläufe	☐	☐	☐
6. Automatische Prüfung	Prüfung von Inhalt (Längen, Recht-schreibung), Layout (Normen, Golden Rules, Styleguide) und Bedien-ablauf (Konsistenz, selbst definierte Regeln)	☐	☐	☐
7. Gemeinsames Arbeiten an Dokumenten	Versionsverwaltung, Freigabe-Management Datenaustausch, gemeinsame Datenbasis, Versions-Management	☐	☐	☐
8. Rollenkonzept	Aufgabenspezifische Benutzerober-fläche, Rollenspezifische Sichten	☐	☐	☐
9. Layoutmöglichkeit von Bildschirmen und Blenden	Zeichenfunktion (Schriften, Farbe, 3D-Effekte, Transparenz ...)	☐	☐	☐

Anforderungen	Beschreibung	weniger wichtig	wichtig	sehr wichtig
Automatisierte Ausgabe				
1. Spezifikation	Übergabe Zulieferer Standardformate	☐	☐	☐
2. Export	Standardformate (doc, pdf …)	☐	☐	☐
3. Testautomatisierung	Vergleich von Spezifikation und Muster bzw. Endgerät	☐	☐	☐
4. Usability-Tests	Probandentests	☐	☐	☐
5. Simulation, virtuelle Prototypen	Look and Feel, Bedienabläufe, Anbindung an CAN/MOST	☐	☐	☐
Technik				
1. Prozesssicherheit	Fehlertoleranz, Reifegrad	☐	☐	☐
2. Einsatz von Standards	Datenformate, Programmiersprachen	☐	☐	☐
3. Support	Hotline, Handbuch	☐	☐	☐
4. Schnittstellen zu anderen Tools	Word, Excel, PowerPoint, CAD, DeltaGen …	☐	☐	☐

C. Bisherige Vorgehensweise

1. Wie beurteilen Sie die aktuelle Vorgehensweise und Toolunterstützung im Entwicklungsprozess?

☐ Gut

☐ Neutral

☐ Verbesserungsfähig

Verbesserungsvorschläge:

2. Welche DV-Werkzeuge verwenden Sie momentan im Entwicklungsprozess?

Programm	**Einsatzzweck, Anwendungsbereiche**
☐ Microsoft Word	
☐ Microsoft PowerPoint	
☐ Microsoft Excel	
☐ Microsoft Visio	
☐ Bildbearbeitungsprogramm	
☐ MindManager	

☐ Sonstiges:

3. Welche Funktionen in den oben genannten Programmen sind in Ihrer täglichen Arbeit besonders hilfreich oder welche Oberflächeneigenschaften gefallen Ihnen besonders gut?

Funktion	**Programm**
Bsp. einfache Zeichenmöglichkeit	PowerPoint

4. Welche Funktionen die in den oben genannten Programmen <u>nicht</u> vorhanden sind, würden Ihnen im Entwicklungsprozess helfen?

Bsp. Versionsverwaltung, Zentrale Datenhaltung

Für eventuelle Rückfragen stehen wir Ihnen gerne zur Verfügung unter der Nummer 88112 oder per E-Mail (tobias.schwarz@porsche.de, mathias.heilig@porsche.de)!

Vielen Dank für Ihre Mitarbeit!

Anhang B

Werkzeuge zur Unterstützung von Usability-Entwicklungsprozessen

Phase	GENIUS	JANUS	MUSEII	EXPOSE	ProUse	Ergo Conceptor	ErgoVal	KRI/AG
Vorentwicklung			Zweckgebunden Sicht Aufnahme der Analyseergebnisse	Sammlung bisheriger Erfahrungen work objects, related methods,	Ablegen bisherigen Erfahrungen			
Serienentwicklung Oberflächendesign	Ergonomische Regelbasis für kompletten Prototype	Regelbasis für Layout	Präsentations-bezoge ne Sicht konkrete Gestaltung der Oberfläche	Präsentations-bezogene Sicht konkrete Gestaltung der Oberfläche	Ablegen bisherigen Erfahrungen	Prüfung Anordnung und Konsistenz	Prüfung Anordnung und Farbgebung	Prüfung Anordnung
Serienentwicklung Informationsdesign	Ergonomische Regelbasis für kompletten Prototype	Keine Angaben	Keine Angaben	Keine Angaben	Ablegen bisherigen Erfahrungen		Prüfung Beschriftung und Bedienelement	Prüfung Beschriftung Bedienelement
Serienentwicklung Interaktionsdesign	Ergonomische Regelbasis für kompletten Prototype	Regelbasis für Menüstrukturen	Interaktionsbezogene Sicht Festlegung der dynamischen Struktur	Interaktionsbezogene Sicht Festlegung der dynamischen Struktur	Ablegen bisherigen Erfahrungen	Transitionen mit Petrinetzen		
Serienbetreuung					Ablegen bisherigen Erfahrungen			

Phase	Sherlock	KAWA-RESMI	ISO Evaluator	AIDE	IDA	GUIDEAS	Interface Ratgeber	GUIDE
Vorentwicklung					Ablegen bisherigen Erfahrungen in der Wissensdatenbank	Assistenzsystemen für Anforderungs-analyse für Bediensysteme		
Serienentwicklung Oberflächendesign	Prüfung Konsistenz Anordnung, und Farbgebung	Prüfung Farbgebung	Konformitätsprüfung nach Norm	Prüfung Anordnung	Prüfung Konsistenz und Anordnung Ablegen bisherigen Erfahrungen in der Wissensdatenbank	Assistenzsystem für Gestaltung der Benutzeroberfläche	Wissensbasis für Gestaltungsregeln	Erfahrungswerte durch vorherige Projekte daraus Empfehlung für ein Anwendungsproblem
Serienentwicklung Informationsdesign	Prüfung Beschriftung Bedienelement	Prüfung Beschriftung Bedienelement	Konformitätsprüfung nach Norm		Prüfung Beschriftung Bedienelement	Assistenzsystem für Gestaltung des Informationsdesigns	Wissensbasis für Gestaltungsregeln	Erfahrungswerte durch vorherige Projekte daraus Empfehlung für ein Anwendungsproblem
Serienentwicklung Interaktionsdesign			Konformitätsprüfung nach Norm	Berücksichtigt die Wege welche vom Cursor zurückgelegt wurden	Buttons müssen beispielsweise spezielle Aktionen auslösen	Assistenzsystem für Gestaltung des Interaktionsdesigns	Wissensbasis für Gestaltungsregeln	Erfahrungswerte durch vorherige Projekte daraus Empfehlung für ein Anwendungsproblem
Serienbetreuung								

Anhang C

Beispiel XAML

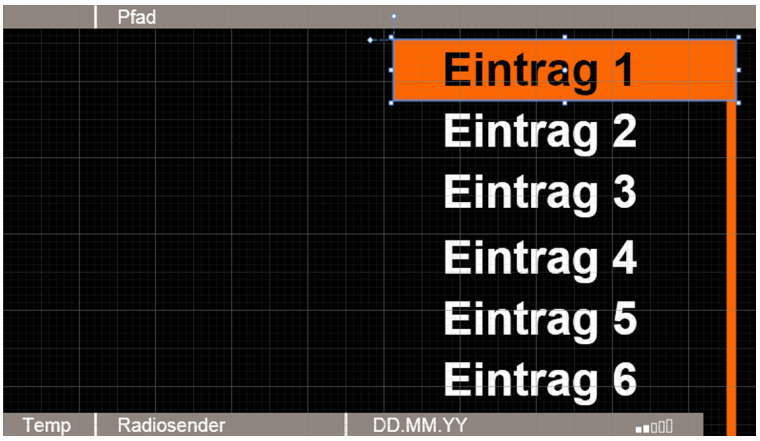

Abbildung C.1: Beispiel PCM Bildschirm

```
<<Window
 xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
 xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
 x:Class=".Window2"
 x:Name="Bildschirm"
 Title="Window2"
 Width="800" Height="480">
 <Grid x:Name="LayoutRoot">
  <Rectangle Fill="#FF000000" Stroke="#FF000000"/>
  <Rectangle Fill="#FFFF6600" Stroke="#FFFF6600" HorizontalAlignment="
      Right" Margin="0,36.5,21.5,0" VerticalAlignment="Top" Width="360"
      Height="64"/>
  <TextBox HorizontalAlignment="Right" Margin="0,39.5,0,0"
      VerticalAlignment="Top" Width="333.5" Height="64" Background="{
      x:Null}" BorderThickness="0,0,0,0" FontFamily="Arial" FontSize="48"
       FontWeight="Bold" Foreground="#FF000000" Text="Eintrag 1&amp;#xd;&amp;#xa;
      " TextWrapping="Wrap"/>
  <TextBox Width="333.5" Height="64" Background="{x:Null}"
      BorderThickness="0,0,0,0" FontFamily="Arial" FontSize="48"
      FontWeight="Bold" Foreground="#FFFFFFFF" Text="Eintrag 2&amp;#xa;"
      TextWrapping="Wrap" HorizontalAlignment="Right" Margin="0,103.5,0,0
      " VerticalAlignment="Top"/>
```

```
<TextBox Width="333.5" Background="{x:Null}" BorderThickness="0,0,0,0"
    FontFamily="Arial" FontSize="48" FontWeight="Bold" Foreground="#
    FFFFFFFF" Text="Eintrag 3&#xa;" TextWrapping="Wrap"
    HorizontalAlignment="Right" Margin="0,167.5,0,216.5"/>
<TextBox Width="333.5" Background="{x:Null}" BorderThickness="0,0,0,0"
    FontFamily="Arial" FontSize="48" FontWeight="Bold" Foreground="#
    FFFFFFFF" Text="Eintrag 4&#xa;" TextWrapping="Wrap"
    HorizontalAlignment="Right" Margin="0,0,0,152.5" VerticalAlignment=
    "Bottom" Height="64"/>
<TextBox Width="333.5" Background="{x:Null}" BorderThickness="0,0,0,0"
    FontFamily="Arial" FontSize="48" FontWeight="Bold" Foreground="#
    FFFFFFFF" Text="Eintrag 5&#xa;" TextWrapping="Wrap"
    HorizontalAlignment="Right" Margin="0,0,0,88.5" VerticalAlignment="
    Bottom" Height="64"/>
<TextBox Width="333.5" Background="{x:Null}" BorderThickness="0,0,0,0"
    FontFamily="Arial" FontSize="48" FontWeight="Bold" Foreground="#
    FFFFFFFF" Text="Eintrag 6&#xa;" TextWrapping="Wrap"
    HorizontalAlignment="Right" Margin="0,0,0,24.5" VerticalAlignment="
    Bottom" Height="64"/>
<Rectangle Fill="#FFFF6600" Stroke="#FFFF6600" RadiusX="0" RadiusY="0"
    HorizontalAlignment="Right" Margin="0,99.5,21.5,0" Width="10"/>
<Rectangle Fill="#FF90867F" Stroke="#FF90867F" RadiusX="0" RadiusY="0"
    x:Name="titlebar" VerticalAlignment="Top" Height="25"/>
<Rectangle Fill="#FF90867F" Stroke="#FF90867F" RadiusX="0" RadiusY="0"
    Margin="0,0,56,0" VerticalAlignment="Bottom" Height="25"/>
<TextBox Background="{x:Null}" BorderThickness="0,0,0,0" FontFamily="
    Arial" FontSize="20" FontWeight="Normal" Foreground="#FFFFFFFF"
    Text="Temp" TextWrapping="Wrap" HorizontalAlignment="Left" Margin="
    17.5,0,0,-0.5" VerticalAlignment="Bottom" Width="62.5" Height="24.5
    "/>
<TextBox Background="{x:Null}" BorderThickness="0,0,0,0" FontFamily="
    Arial" FontSize="20" FontWeight="Normal" Foreground="#FFFFFFFF"
    Text="Radiosender" TextWrapping="Wrap" Width="122.688" Height="24.5
    " HorizontalAlignment="Left" Margin="117.312,0,0,-0.5"
    VerticalAlignment="Bottom"/>
<TextBox Background="{x:Null}" BorderThickness="0,0,0,0" FontFamily="
    Arial" FontSize="20" FontWeight="Normal" Foreground="#FFFFFFFF"
    Text="DD.MM.YYY" TextWrapping="Wrap" Height="24.5" Margin="
    385.5,0,288,0" VerticalAlignment="Bottom"/>
<Rectangle Fill="#FFFFFFFF" Stroke="#FFFFFFFF" HorizontalAlignment="
    Left" Margin="96,0,0,0" VerticalAlignment="Bottom" Width="2.75"
    Height="25"/>
<Rectangle Fill="#FFFFFFFF" Stroke="#FFFFFFFF" Width="2.75" Height="25"
     HorizontalAlignment="Left" Margin="359.271,0,0,0"
    VerticalAlignment="Bottom"/>
<Rectangle Fill="#FFFFFFFF" Stroke="#FFFFFFFF" Width="6" Height="6"
    HorizontalAlignment="Right" Margin="0,0,121,4.25" VerticalAlignment
    ="Bottom"/>
<Rectangle Fill="#FFFFFFFF" Stroke="#FFFFFFFF" Width="6" Height="8"
    HorizontalAlignment="Right" Margin="0,0,113,4.25" VerticalAlignment
    ="Bottom"/>
<Rectangle Fill="#FF90867F" Stroke="#FFFFFFFF" Width="6" Height="10"
    HorizontalAlignment="Right" Margin="0,0,104.75,4.25"
```

```
      VerticalAlignment="Bottom"/>
  <Rectangle Fill="#FF90867F" Stroke="#FFFFFFFF" Width="6" Height="12"
      HorizontalAlignment="Right" Margin="0,0,96.75,4.25"
      VerticalAlignment="Bottom"/>
  <Rectangle Fill="#FF90867F" Stroke="#FFFFFFFF" Width="6" Height="14"
      HorizontalAlignment="Right" Margin="0,0,88.75,4.25"
      VerticalAlignment="Bottom"/>
  <Rectangle Fill="#FFFFFFFF" Stroke="#FFFFFFFF" Width="2.75"
      HorizontalAlignment="Left" Margin="96,0,0,0" VerticalAlignment="Top
      " Height="25"/>
  <TextBox Background="{x:Null}" BorderThickness="0,0,0,0" FontFamily="
      Arial" FontSize="20" FontWeight="Normal" Foreground="#FFFFFFFF"
      Text="Pfad" TextWrapping="Wrap" Width="118.5" HorizontalAlignment="
      Left" Margin="117.312,0.455,0,0" x:Name="path" VerticalAlignment="
      Top" Height="24.5"/>
 </Grid>
</Window>
```

Listing C.1: Quelltext-XAML Bildschirm Menüeineintrag rechts

Anhang D

MetaEdit+ Report

```
Report

/* generate java classes */
foreach .(*Bildschirm*) {

  filename; :Name; '_' oid; '.java'; write;
  ...
  'public ScreenInterface getScreenAfterEvent(String event) {'newline;
    'ScreenInterface retval = super.getScreenAfterEvent(event);'newline;
    ...
    subreport; '_getScreenAfterEvent'; run;
    ...
  '}'
  ...
  close;
}
...
endreport


Report '_getScreenAfterEvent'

do ~Von~Nach
   {
  if > 1 HauptbedienTaste:Hauptbedientaste = 'MAIN' then 'if (event ==
      EventDefines.TEL)'; endif
  ...
  newline;
  if > 1 HauptbedienTaste; then
  do .() {
  '    retval = (ScreenInterface) ScreenMap.getInstance().get("' :Name; '
      _' oid; '");'
  }
  endif;
}
endreport
```

Listing D.1: Quelltext Report