Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Thomas Schmidt

Ontologiebasierte, adaptive Mitteilungsverarbeitung

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Thomas Schmidt

Ontologiebasierte, adaptive Mitteilungsverarbeitung

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang Informatik
am Studiendepartment Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. Jörg Raasch

Abgegeben am 19. März 2008

Thomas Schmidt

Thema der Masterarbeit

Ontologiebasierte, adaptive Mitteilungsverarbeitung

Stichworte

Benutzerprofil, Ontologie, Offenes Benutzermodell

Kurzzusammenfassung

Software-Programme sollen den Benutzer so gut wie möglich bei der Erfüllung seiner Aufgaben unterstützen. Benutzer haben unterschiedliche Charakteristika, aus denen unterschiedliche Anforderungen an ein und dieselbe Software hervorgehen. Eine Möglichkeit auf die individuellen Anforderungen einzugehen, sind adaptive Programme, die sich den Bedürfnissen der Nutzer entsprechend anpassen. Zur Repräsentation von Wissensbereichen werden in der Informatik Ontologien verwendet. In dieser Arbeit werden diese beiden Bereiche anhand eines Anwendungsszenarios für die Vermittlung von Mitteilungen miteinander verknüpft. Dabei wird untersucht, ob ein transparentes Benutzermodell mit ontologiebasierten Strukturen und ontologiebasierter Navigation einen Mehrwert für den Nutzer des Systems erbringt. Zu diesem Zweck wird ein Anwendungssystem entworfen und prototypisch implementiert. Um die Zielerreichung zu evaluieren werden Usability-Untersuchungen durchgeführt.

Thomas Schmidt

Title of the paper

Ontology-based adaptive message processing

Keywords

User profile, ontology, open user model

Abstract

With respect to task performances, software programs are supposed to support their users in the best possible way. Different users, however, also have different characteristics, which results in a whole range of varied requirements for one and the same software. In this context, adaptive programs responding to their users' respective needs constitute a possibility to comply with such individual requirements. In computer science, ontologies are utilised for the representation of different fields of knowledge. Against this background, this thesis combines these two fields on the basis of an application scenario for message communication. In doing so, the question whether a transparent user model with ontology-based structures and navigation creates an additional benefit for the user of the system is analysed. For this purpose, an application system is designed and implemented in a prototypical manner. In order to evaluate the related target achievement, a number of usability analyses are conducted.

Inhaltsverzeichnis

1 Einleitung

Software-Programme sollen den Benutzer so gut wie möglich bei der Erfüllung seiner Aufgaben unterstützen. Benutzergruppen und auch die Benutzer innerhalb einer Gruppe haben unterschiedliche Charakteristika aus denen unterschiedliche Anforderungen an ein und dieselbe Software hervorgehen. Adaptive Software ermöglicht es, auf diese individuellen Anforderungen einzugehen. Das Benutzerprofil kann angepasst und die Funktionalität des Systems für den Benutzer optimiert werden.

1.1 Motivation

Bei der großen Menge an Informationsquellen fällt es Benutzern zunehmend schwer Informationen aufzufinden und einen Überblick zu behalten. Hinzu kommt, dass die gefundenen Inhalte unterschiedlich wahrgenommen werden. Eine Eye-Tracking-Studie der Firma seo-consulting zur Suchmaschinennutzung ergab:

„*Die ersten drei Plätze in den Suchergebnissen werden von 100% aller Suchenden gelesen, der zehnte Platz hingegen nur noch von 20% der Besucher.*“ [SC05]

Es werden Strategien benötigt um die für einen Benutzer relevante Informationen zu finden. Eine Strategie sind Benutzerprofile, mit deren Hilfe die Informationen entsprechend gefiltert und somit reduziert werden [CCCJ04].

Adaptive Programme können sich Bedürfnissen und Eigenschaften eines Benutzers anpassen. Somit wird die Funktionalität optimiert und der Benutzer besser bei der Erreichung seiner Anforderungen unterstützt. Einige Projekte haben sich bereits mit solchen anpassungsfähigen Systemen beschäftigt. Deren Prozesse sind meist kompliziert und für den Benutzer nur wenig transparent [ABG+07].

Benutzer haben mentale Modelle über die Funktionsweise von Systemen. Diese beeinflussen die Art und Weise wie sie Situationen interpretieren und welche Entscheidungen sie treffen [GRG04]. Wenn ein Benutzer diese Modelle nicht aufbauen kann, kann dies zu Frustration und geringem Vertrauen in das System führen. Ein Ansatz zur Lösung des Problems sind „open user model“-Systeme. Hierbei steht die Transparenz im Vordergrund. Der Benutzer kann, das Benutzermodell ändern und kontrollieren [ABG+07].

Wenn ein Nutzer seine Erfahrungen in einem Wissensbereich berücksichtigt sieht, kann dies sein Verständnis über die Funktionsweisen und die Steuerung des Systems erhöhen. Diese

konzeptionelle Formalisierung von Wissensbereichen kann in der Informatik mit Ontologien erreicht werden. Mit einer Ontologie soll ein formalisiertes Modell der Welt oder eines Teils der Welt dargestellt werden. Die Begriffe und Zusammenhänge sollen für den Menschen aufgrund seines Verständnisses der Welt implizit klar sein. Durch klar abgegrenzte Ontologien mit möglichst einfachen Konzepten soll ein gemeinsames Konzept über ein Wissensgebiet entstehen [Gru93].

Neben dem Inhalt ist die Navigation der wichtigste Bestandteil einer Website. Lt. Steve Krug werden Personen eine Website nicht benutzen, wenn sie sich darin nicht zurechtfinden [Kru06]. Der Benutzer sollte also klar erkennen, wie die Navigation funktioniert. Sie sollte möglichst einfach und konsistent sein. Dies gilt ebenso für die Navigation mit dem Benutzermodell.

Um den Benutzer möglichst gut beim Auffinden relevanter Informationen zu unterstützen und eine hohe Transparenz zu erreichen, werden in dieser Arbeit die Eigenschaften des „open user models“ mit einem ontologiebasierten Benutzerprofil und ontologiebasierter Navigation vereint. Durch das Modell der Ontologien soll dem Benutzer die Navigation und Kontrolle innerhalb des Benutzermodells erleichtert werden. Der Benutzer soll in die Lage versetzt werden ein mentales Modell der Anwendung zu erstellen, das auf seinem bereits vorhandenen Verständnis der Welt beruht.

1.2 Ziele der Arbeit

In dieser Arbeit soll ein neuer Ansatz im Bereich der Mitteilungsverarbeitung im Umfeld des Web 2.0 untersucht werden. Dieser beruht auf einem ontologiebasierten, offenen Benutzerprofil („open user model“) und ontologiebasierter Navigation. Moderne Webtechnologien werden mit aktuellen Usability-Aspekten vereint, um dem Benutzer einen komfortablen Zugang zu Mitteilungen und eine intuitive Benutzung des Systems zu ermöglichen. Das Benutzerprofil soll vom Benutzer frei verändert und kontrolliert werden können. Der Benutzer kann durch diese Änderungen Einfluss auf die für ihn relevanten Mitteilungen nehmen und sie nach Bedarf eingrenzen. Nach einer Änderungen erhält er eine direkte Rückmeldung.

Es soll gezeigt werden, wie ein solcher Ansatz das Vertrauen der Benutzer in profilgestützte Systeme erhöhen kann. Zudem soll überprüft werden, ob die transparente Darstellung des Profils und die Navigationsstrukturen einen Nutzen für den Benutzer darstellen.

Für die Evaluierung wird ein Software-Entwurf und eine prototypische Implementierung erstellt. Anhand des Prototypen werden die Anwendungsfälle betrachtet und Benutzerstudien durchgeführt.

1.3 Aufbau der Arbeit

Die Vorgehensweise der Arbeit orientiert sich sowohl am Unified Process als auch am Ansatz des User Centered Design. Der Unified Process bildet das Rahmenwerk für den gesamten Entwicklungsprozess und gliedert die Arbeit in die Kapitel Systemanalyse, Systementwurf, Systemimplementierung und Systemevaluierung. Dieser Ansatz wurde gewählt, da er ein erfolgreiches Prozessmodell zur Entwicklung objektorientierter Software ist und auf der Basis von so genannten „Best Practices“ entwickelt wurde [RG02]. Der User Centered Design-Ansatz wurde gewählt, um eine einfach zu bedienende Software zu entwickeln, die den Benutzer bei seinen Aktivitäten unterstützt und ihm einen Mehrwert bietet [KH98].

Im Anschluss an die Einleitung werden in Kapitel 2 die, für diese Arbeit relevanten, Grundlagen erläutert. Zunächst wird in Abschnitt 2.1 dargestellt was unter einem Benutzerprofil zu verstehen ist. In Abschnitt 2.2 wird der Begriff und die Struktur von Ontologien beschrieben. In Abschnitt 2.3 wird die Ajax-Technologie beleuchtet und in diesem Zusammenhang Bezug auf das Google Web Toolkit (GWT) genommen. Abschließend diskutiert Abschnitt 2.4 das Thema Privacy.

Kapitel 3 stellt zu Beginn in Abschnitt 3.1 ein Ausgangsszenario für diese Arbeit vor, auf dessen Grundlage das System entworfen wird. In Abschnitt 3.2 werden Projekte vorgestellt, die sich in angrenzenden Bereichen dieser Arbeit bewegen. Abschnitt 3.3 stellt die Vorgehensweise zur Ermittlung und die verwendete Technik der Anforderungsanalyse dar. Die funktionalen Anforderungen werden in Abschnitt 3.4 durch Anwendungsfälle beschrieben und die daraus abgeleiteten Systemabläufe vorgestellt. Abschließend beschreibt Abschnitt 3.5 die nichtfunktionalen Anforderungen der Anwendung.

Im Anschluss an die Analyse wird in Kapitel 4 die Architektur des entwickelten Systems diskutiert. Abschnitt 4.1 stellt die Server- und Abschnitt 4.2 die Client-Architektur vor.

Kapitel 5 beschreibt die Anwendung aus Implementierungssicht. Zunächst wird in Abschnitt 5.1 die prototypische Implementierung für den Server- und Client-Teil der Anwendung beschrieben. Anschließend werden in Abschnitt 5.2 die zur Entwicklung verwendeten Software-Komponenten vorgestellt. Abschnitt 5.3 beschreibt die eingesetzten Maßnahmen zur Qualitätssicherung der Implementierung.

In Kapitel 6 wird eine Evaluierung des entwickelten Systems beschrieben. Hierzu werden in Abschnitt 6.1 die Versuchsziele und Erwartungen diskutiert. Die Versuchsdurchführung der Usability-Untersuchungen wird in Abschnitt 6.2 vorgestellt. Hierbei wird auf den Umfang und den Ablauf der Tests sowie auf die verwendeten Hard- und Softwarekomponenten eingegangen. Abschließend werden die Versuchsergebnisse in Abschnitt 6.3 dargestellt.

Als Abschluss fasst Kapitel 7 die Erkenntnisse dieser Arbeit in Abschnitt 7.1 zusammen und bewertet diese in Abschnitt 7.2. Abschnitt 7.3 gibt einen Ausblick für mögliche Verbesserungen und Weiterentwicklungen.

2 Grundlagen

In diesem Kapitel werden verschiedene Grundlagen aufgezeigt, die für das Verständnis der Arbeit relevant sind. Abschnitt 2.1 erläutert, was unter einem Benutzerprofil zu verstehen ist. Abschnitt 2.2 beschreibt den Begriff und die Struktur von Ontologien und geht auf die in dieser Arbeit verwendete Web Ontology Language ein. In Abschnitt 2.3 wird erklärt, in welchem Bereich Ajax anzusiedeln ist und was man unter dieser Technologie versteht. In diesem Zusammenhang wird in Abschnitt 2.3.1 Bezug auf das Google Web Toolkit (GWT) genommen. Abschließend wird in Abschnitt 2.4 das Thema Privacy diskutiert.

2.1 Benutzerprofil

Der Begriff „Benutzerprofil“ wird in der Literatur nicht einheitlich verwendet. Im Usability Engineering werden Benutzerprofile (englisch „User Profiles“) erstellt um Nutzergruppen zu beschreiben, die unterschiedliche Anforderungen an ein System haben [CR03]. Bei der Personalisierung von Systemen wird der Begriff mit „Interest-Profile“ geleichgesetzt. Ein solches Profil repräsentiert die Langzeitinteressen eines Benutzers [BC92]. In den beiden nachfolgenden Abschnitten werden Benutzerprofile im Kontext des Usability Engineerings sowie der Personalisierung betrachtet.

2.1.1 Usability Engineering

Im Usability-Engineering werden Benutzer anhand ihrer Charakteristika gruppiert. Es werden nur Eigenschaften von Benutzern miteinbezogen, die Auswirkungen auf die Gestaltung eines Systems haben [May99].

Ein Benutzerprofil beschreibt den Nutzer hinsichtlich folgender Merkmale:

- Psychologische Charakteristika (z.B. Gesinnung, Motivation)
- Wissensstand und Erfahrung (z.B. Computerkenntnisse, Aufgaben Erfahrung)
- Tätigkeit und Aufgaben Charakteristika (z.B. Benutzungshäufigkeit, Aufgabenstruktur)
- Physische Charakteristika (z.B. Farbenblindheit)

Ziel dieser Methode ist es Anforderungen abzuleiten, die für bestimmte Gruppen von Nutzern gelten.

Ein benutzerfreundliches System muss die Anforderungen der Benutzer erfüllen. Aufgrund deren Heterogenität gibt es kein alleiniges Interface für alle Benutzern. Benutzergruppen können Aufschlüsse darüber geben, wie das System unterschiedliche Benutzergruppen adressiert. Sind die Benutzer heterogen, kann es notwendig sein, unterschiedliche User Interfaces zu entwickeln, die auf die Bedürfnisse der jeweiligen Gruppe zugeschnitten sind [May99].

Um Benutzergruppen und ihre Anforderungen an ein System zu veranschaulichen, werden „Personas“ erstellt. Eine Persona ist eine fiktive Person mit den typischen, konkreten Eigenschaften einer Benutzergruppe. Personas veranschaulichen eine Sammlung von Charakteristika und erleichtern die Kommunikation über Benutzerprofile.

Benutzerprofile und Personas werden im Usability Engineering im Rahmen der Kontextanalyse ermittelt.[1]. Unter anderem sind Fokusgruppen, Feldbeobachtungen, Interviews oder Befragungen Methoden, anhand derer man Erkenntnisse über die Eigenschaften von Benutzern gewinnt. Sie geben Rückschluss darauf, welchen Anforderungen das System genügen muss und welche Funktionen bereitgestellt werden müssen [CR03].

2.1.2 Personalisierung

Im Bereich der Personalisierung wird das Benutzerprofil verwendet, um die angebotenen Dienste bzw. deren Inhalte individuell für den jeweiligen Benutzer aufzubereiten.

Es werden verschiedene Informationen über den Benutzer benötigt, die oftmals vertraulich sind.[2] Neben allgemeinen Angaben zur Person werden deren Interessen aufgenommen, die verschiedenen Zwecken dienen können.

Collaborative Filtering: Die Grundidee von Collaborative Filtering Systemen ist, Personen miteinander zu vergleichen und so aus den Präferenzen ähnlicher Personen Rückschlüsse auf die Präferenzen einer bestimmten Person zu ziehen. Die gesammelten Benutzerdaten werden auf Ähnlichkeiten untersucht, um Prognosen zur Individualisierung von Inhalten abzuleiten. Den Benutzern können so Vorschläge unterbreitet oder Hinweise hinterlassen werden [Dör03].

Information Filtering: Typische Information Filtering Systeme arbeiten mit einer großen Anzahl von Texten oder anderen Medien. Die Benutzerdaten werden verwendet um die Datenmenge einzugrenzen. Somit bekommt der Benutzer nur die Daten angezeigt, die für ihn von Interesse sind [BC92].

[1] Der Begriff „Persona“ wir in der Praxis oftmals synonym für dem Begriff „User Profil“ verwendet [CR03]
[2] Für nähere Informationen siehe Kapitel 2.4

2.2 Ontologie

Diese Arbeit verwendet Ontologien für die Repräsentation spezieller Themenbereiche und benutzerspezifischer Interessen. In diesem Abschnitt wird der Begriff „Ontologie“ näher erläutert und auf aktuelle Entwicklungen in diesem Bereich eingegangen, die für diese Arbeit eine Rolle spielen.

2.2.1 Begriff

Der Begriff *Ontologie* kommt ursprünglich aus der Philosophie und bedeutet „die Lehre vom Sein“. Er bezeichnet die Wissenschaft der Beschreibung von Arten und Dingen und deren Beziehungen zueinander.

In der Informatik ist mit einer Ontologie eine konzeptuelle Formalisierung von Wissensbereichen und Begriffssystemen gemeint. Diese Wissensrepräsentation kapselt Wissen über eine bestimmte Domäne und beschreibt deren Konzepte und Relationen.

„*An ontology is a formal, explicit specification of a shared conceptualization.*“ [T.G93]

Ontologien haben durch die Idee des „Semantic Web“ einen Aufschwung erlebt. Diese Idee stammt von Tim Berners-Lee, dem Begründer des heutigen Web: Das Semantic Web soll es ermöglichen, Daten zwischen Maschinen auszutauschen [HL04].

Die aktuellste Entwicklung der standard Ontologiesprachen ist OWL (Web Ontology Language)[3] vom World Wide Web Consortium (W3C) [W3C04a].

2.2.2 RDF

RDF steht für Resource Description Framework. RDF ist eigentlich eine XML-Sprache, mit der Bedeutung ausgedrückt werden soll. Eine Ressource, die innerhalb des Dokuments durch eine URI identifiziert wird, bekommt mit diesem Konzept eine Bedeutung zugeordnet [W3C04b].

Der Aufbau eines RDF-Dokumentes beginnt mit dem Wurzelelement *<RDF>*, in dem auch die Namespaces angegeben werden. Das *<Description>*-Element enthält die eigentlichen Informationen über die Ressource. Das *about*-Attribut gibt die URI der Ressource an und die darauffolgenden Elemente sind Beschreibungen mit eigenen Werten.

Ein RDF-Modell besteht aus drei Teilen: Subjekt, Prädikat und Objekt. Diese sind aus der Grammatik von Sprachen übernommen und bilden die Beschreibung des Subjekts.

Ein RDF-Dokument, dass in XML serialisiert wurde, heißt auch RDF/XML. Zur Vereinfachung der Modellierung gibt es sowohl grafische Werkzeuge[4] als auch die Sprache N3 (Notation 3) [W3C06].

[3] Siehe Kapitel 2.2.4

[4] In der Arbeit wurde der Ontologie Editor Protégé [Sta07] verwendet.

2.2.3 RDF-Schema

RDF-Schema ergänzt RDF um die Möglichkeit, ein eigenes RDF-Vokabular zu erstellen. Ein solches Vokabular kann als gemeinsame Basis für Anwendungen verwendet werden [W3C04c].

Ein bekanntes Beispiel für ein RDF-Vokabular ist z.B. das FOAF (Friend of a Friend)-Vokabular [Ben07].[5]

2.2.4 OWL

OWL steht für Web Ontology Language. Eine Ontologie im Sinne von OWL besteht aus der Klassifizierung von Objekten und aus der Abbildung von deren Beziehungen und Eigenschaften. Das Ziel dieser Ontologie ist, einen Wissensbereich mit all seinen Beziehungen abzubilden [W3C04a].

Eine OWL-Ontologie besteht aus Klassen, Eigenschaften und Instanzen. Durch diese Struktur lässt sich ein semantisches Netz aufbauen, bei dem die Klassen und Instanzen die Knoten und die Eigenschaften die Kanten bilden. Technisch gesehen ist diese Konstruktion eine Erweiterung von „RDF“ (Recource Description Framework)[6], geht aber darüber hinaus. OWL-Dokumente können in RDF/XML geschrieben werden und enthalten eine oder mehrere Ontologien. Sie definieren gemäß „RDF Schema“[7] ein Vokabular, das zum Beschreiben der Ontologien dient.

2.2.4.1 Versionen

Es gibt drei verschiedene OWL-Sprachen: *OWL Lite*, *OWL DL* und *OWL Full*. Diese sind nach Ausdrucksfähigkeit gestaffelt:[8]

- **OWL Lite** ist die syntaktisch einfachste Sprache. Sie sollte für einfache Klassenhierarchien und einfache Randbedingungen gebraucht werden. OWL Lite hat eine geringere formale Komplexität als OWL DL[9] und ist eine Teilmenge von OWL DL.
- **OWL DL** ist ausdrucksstärker als OWL Lite und basiert auf *Description Logics*. Die Beschreibungslogik ist eine Untermenge der Prädikatenlogik, und sie ist entscheidbar. Es ist möglich, Schlussfolgerungen zu ziehen und Inkonsistenzen in einer, auf OWL DL basierenden, Ontologie zu finden. OWL DL ist eine Teilmenge von OWL Full.

[5] Eine Spezifikation zum FOAF-Vokabular ist unter [BM04] zu finden.
[6] Siehe Kapitel 2.2.2
[7] Siehe Kapitel 2.2.3
[8] Eine detailliertere Übersicht ist unter: [W3C04a] zu finden.
[9] Für weitere Details siehe http://www.w3.org/TR/owl-ref/#OWLLite

- **OWL Full** ist die ausdrucksstärkste OWL-Untersprache. OWL Full ermöglicht das vordefinierte Vokabular zu erweitern, was die Entscheidbarkeit durch eine Schlussfolgerungs-Software erschweren kann. Laut [W3C04a] ist es unwahrscheinlich, dass irgendeine Schlussfolgerungs-Software das komplette Schlussfolgern für jedes Feature von OWL Full unterstützen wird. Diese Version sollte für Situationen verwendet werden, in denen Aussagekraft wichtiger ist als die Entscheidbarkeit.

Abbildung 2.1 visualisiert die Ausdrucksmächtigkeit der verschiedenen OWL-Sprachen.

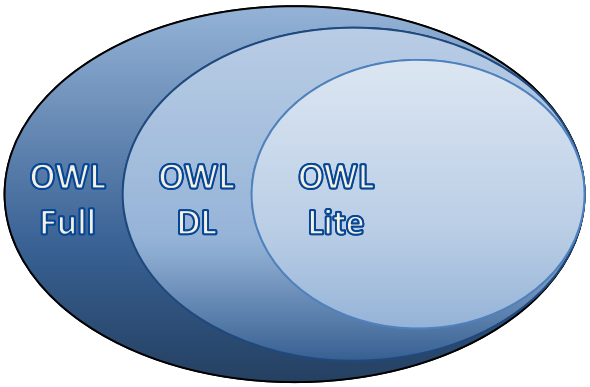


Abbildung 2.1: Übersicht über die verschiedenen OWL-Sprachen

2.2.4.2 Konstrukte

In diesem Abschnitt werden einige für diese Arbeit relevanten OWL-Konstrukte und deren Verwendung erläutert.

Klassen

Eine Klasse definiert in OWL eine Gruppe von Individuals, die eine Eigenschaft gemeinsam haben.

Durch die Verwendung des Tags `<owl:Class>` und dem Attribut `rdf:ID` ist es in OWL möglich benannte Klassen zu definieren.

```
<owl:Class rdf:ID="programmierung"/>
```

Im obigen Beispiel wird eine Klasse *programmierung* definiert, bei der das Attribut `rdf:ID` den Namen der Klasse festlegt. Innerhalb der Ontologie kann diese Klasse anhand des Namens mit vorangestelltem # referenziert werden. In diesem Beispiel: `#programmierung`.

Die Angabe `xmlns="http://www.masterthesis-schmidt.de/Programmierung.owl#"` legt fest, dass sich die Klasse in einer Datei „Programmierung.owl“ auf einer Internetseite „http://www.masterthesis-schmidt.de“ befindet. Durch das Angeben von `http://`

`www.masterthesis-schmidt.de/Programmierung#programmierung` können auch andere Ontologien diese Klasse referenzieren.

Um Hierarchien definieren zu können, wird das Konstrukt `<rdfs:subClassOf>` verwendet. Hiermit kann zum Beispiel ausgedrückt werden, dass `java` eine Unterklasse von `sprachen` ist, und dass `methoden` und `sprachen` Unterklassen von `programmierung` sind.

```
<owl:Class rdf:ID="programmierung"/>
  <owl:Class rdf:ID="java">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="sprachen"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#sprachen">
    <rdfs:subClassOf rdf:resource="#programmierung"/>
  </owl:Class>
  <owl:Class rdf:ID="methoden">
    <rdfs:subClassOf rdf:resource="#programmierung"/>
  </owl:Class>
```

Desweiteren gibt es in OWL zwei spezielle Klassen `<owl:Thing>` und `<owl:Nothing>`. `<owl:Thing>` gibt die oberste und `<owl:Nothing>` die unterste aller Klassen an.

Instanzen

Die Instanzen von OWL-Klassen werden auch „Individuen“ genannt. Ein Individuum kann durch einen Tag mit dem Namen der instanziierten Klasse definiert werden. Dies geschieht mittels des Attributs `rdf:ID`.

```
<java rdf:ID="Java"/>
```

Das Beispiel definiert also ein Individuum `Java`, das der Klasse `java` angehört. Es kann genau wie die Klassen mit Hilfe des #-Zeichens und der dazugehörigen URL referenziert werden.

Eigenschaften

Beziehungen lassen sich in OWL unter Verwendung von Eigenschaften („Properties“) ausdrücken. Hierbei handelt es sich um das Prädikat der in Abschnitt 2.2.2 angesprochenen Subjekt-Prädikat-Objekt-Beziehung.

Ein `DatatypeProperty` stellt eine Beziehung zwischen einem Individuum und einem XML-Schema Datentyp oder einem RDF-Literal her.

```
<owl:DatatypeProperty rdf:ID="hatInteressenWert">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  <rdfs:domain rdf:resource="#programmierung"/>
</owl:DatatypeProperty>
```

Das obige Beispiel definiert die Eigenschaft `hatInteressenWert` und verbindet diese mit dem Datentyp `int`, was durch das Tag `<rdfs:range>` ausgedrückt wird. Wenn dieses Tag nicht angegeben ist, bedeutet dies, dass die Eigenschaft von irgend einem Datentyp (Any) sein kann. Das Tag `<rdfs:domain>` wird angegeben, um die Klassenzugehörigkeit der Instanz zu beschränken. In diesem Beispiel bedeutet dies, dass alle Individuen, die der Klasse bzw. einer Unterklasse von `#programmierung` angehören, diese Eigenschaft besitzen können.

Die definierte Eigenschaft wird instanziiert, indem man bei dem Individuum (Subjekt) ein Tag mit dem Namen der Eigenschaft (Prädikat) einfügt, das auf das Objekt verweist.

```
<java rdf:ID="Java">
  <hatInteressenWert rdf:datatype="&xsd;int">100</hatInteressenWert>
</java>
```

Dieses Beispiel verwendet eine, im `DOCTYPE` definierte Abkürzung `&xsd;int`, die gleichbedeutend mit `http://www.w3.org/2001/XMLSchema#int` ist. Die Instanz Java besitzt die Eigenschaft `hatInteressenWert` der ein Wert von 100 zugewiesen ist.

In OWL gibt es eine Vielzahl weiterer Eigenschaften und Restriktionen, mit denen sich verschiedene Beziehungen ausdrücken lassen. Für weitere Informationen wird an dieser Stelle auf [W3C04a] verwiesen, da diese für diese Arbeit keine weitere Relevanz haben.

Namespaces

Mehrere Dinge können den selben Namen haben. Daher ist es notwendig, dass deren Zusammenhang unterschieden werden kann. Wenn das Wort „Java“ in einem Text vorkommt, sollte ersichtlich sein, ob es sich um die Programmiersprache, die Insel oder den Kaffee handelt. Dieses Problem wird in OWL durch ein Präfix, das einen Namensraum („Namespace“) angibt, gelöst. In OWL werden Präfixe einem Namespace zugeordnet, durch dessen Voranstellung der Name einzigartig wird. Dieser wird durch das Konstrukt `xmlns:` eingeleitet.

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
```

Im obigen Beispiel wird festgelegt, dass sich der Präfix `rdf:` auf die Namespace-Definition unter `www.w3.org/1999/02/22-rdf-syntax-ns#"` bezieht. Hierbei handelt es sich um das RDF-Vokabular. Entsprechendes gilt für die Präfixe `owl:`, `xsd:` und `rdfs:`.

Die Abbildung 2.2 zeigt die in diesem Abschnitt erläuterte Beispiel OWL-Ontologie. Diese wurde mit dem Programm Protégé [Sta07] unter Verwendung des OWLViz-Plugins [Hor04] erstellt.[10]

Die vollständige Beispiel OWL-Ontologie ist im Anhang A.1 zu finden.

[10] Siehe Kapitel 5.2

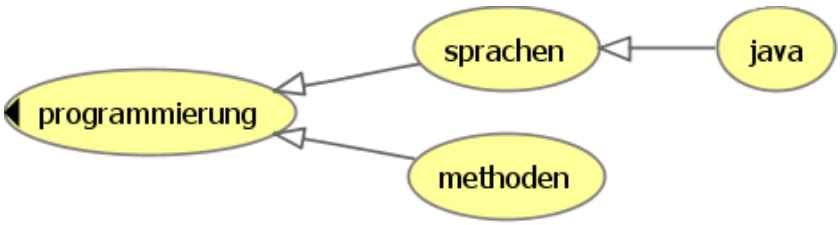


Abbildung 2.2: Beispiel einer OWL-Ontologie

2.2.4.3 Reasoner

Die formale Semantik von OWL ermöglicht es aus einer Ontologie erweiterte Informationen abzuleiten, die nicht explizit enthalten sind [Win93]. Diese werden mit so genannten „Reasoner"-Programmen (Schlussfolgern engl. reasoning) hergeleitet. OWL verhindert keine widersprüchlichen Aussagen. Widersprüche können aber je nach Konstruktion einer Ontologie durch den Reasoner festgestellt werden [PS04].

Im oben genannten Beispiel, das durch die Abbildung 2.2 visualisiert wird, ist `java` eine Unterklasse von `sprachen` und `sprachen` eine Unterklasse von `programmierung`. Das Individuum `Java` ist eine Instanz der Klasse `java`. Ein Reasoner kann daraus schließen, dass `Java` auch ein Individuum von `sprachen` und `programmierung` ist.

In dieser Arbeit wird der Pellet-Reasoner [CP07] verwendet.[11]

2.3 Ajax

Das World Wide Web ist bekannt als zentrales Informationsmedium der modernen Welt. Es war in der ersten Form rein statisch und die Anwender konnten lediglich passiv konsumieren. Das WWW hat sich im Laufe der Zeit mehr und mehr zu einem bidirektionalen Instrument entwickelt, in dem Anwender aktiv in die Gestaltung von Webseiten und deren Inhalte eingreifen können. Die Versionierung Web 2.0 soll genau wie die Versionen bei Programmen den Evolutionssprung verdeutlichen [O'R05]. Zu den Schlüsseltechnologien dieser Weiterentwicklung zählen Ajax (Asynchronous JavaScript and XML), Mashup, SNA (Social Network Analysis) und Kollektive Intelligenz.[12]

Ajax ist die Abkürzung für „Asynchronous JavaScript and XML" und ist ein Konzept zur Entwicklung von Web-Anwendungen [Gar05].

Das Anwendungsprotokoll des World Wide Web ist HTTP (Hyper Text Transfer Protocol), das ein verbindungsloses oder zustandsloses Protokoll ist. Bei dieser Art der Kommunikation mit

[11] Siehe Kapitel 5.2

[12] Mashup, SNA und Kollektive Intelligenz werden hier nicht weiter erläutert, da sie für die Arbeit nicht weiter relevant sind.

einem Server kann dieser nicht erkennen, ob ein Browser bereits vorher eine Anfrage abgeschickt hat. Dies hat zur Folge, dass ein Browser vom Server bei jeder Anforderung von Informationen immer eine vollständig neue Webseite zugeschickt bekommt [KR02].

Ajax ist ein Ansatz, bei dem nicht jede Datenanforderung des Clients das Versenden einer vollständig neuen Webseite notwendig macht. Es ist möglich, nur den Teil einer Webseite vom Server anzufordern, der auch wirklich ausgetauscht werden muss. Die Teile der Webseite, die nicht erneuert werden müssen, können somit erhalten bleiben. Somit wird die zu versendende Datenmenge reduziert und das Antwortverhalten von interaktiven Webapplikationen wird beschleunigt.

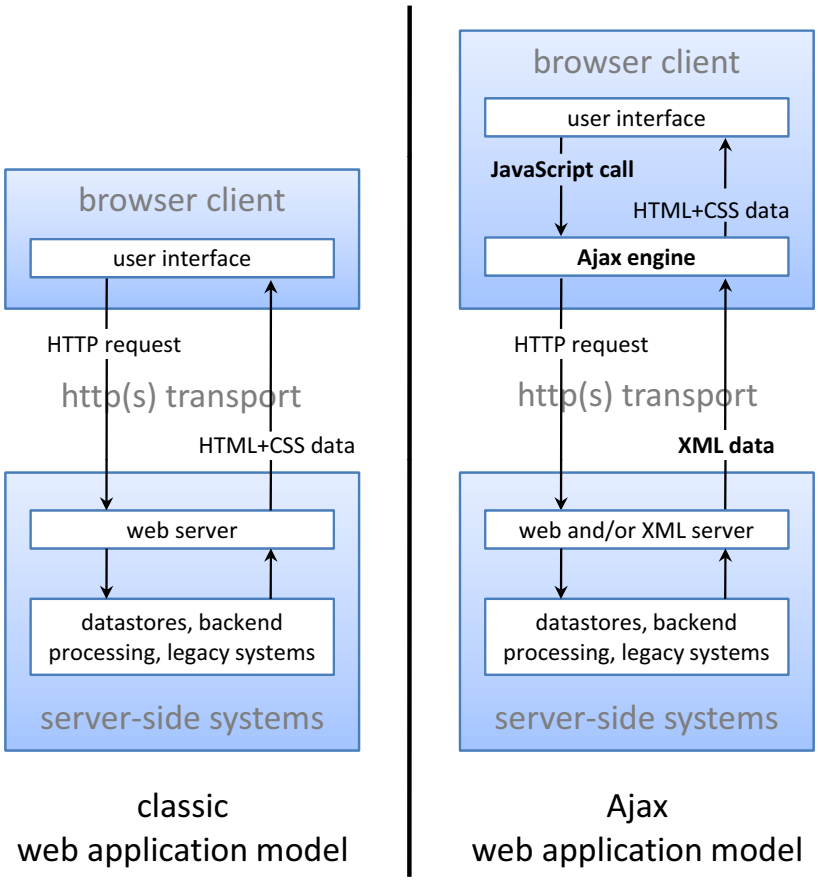


Abbildung 2.3: Vergleich des traditionellen Modells für Web-Anwendungen und dem Ajax-Modell

Abbildung 2.3 zeigt den Vergleich zwischen dem traditionellen Modell für Web-Anwendungen und dem Ajax-Modell [Gar05]. Anstelle einer Webseite läd der Browser zu Beginn einer Session eine Ajax-Engine. Diese ist in JavaScript geschrieben und kümmert sich sowohl um die Darstellung des Interfaces, als auch um die Kommunikation mit dem Server. Diese Vorgänge erfolgen asynchron und somit sind die Interaktion des Users mit der Anwendung und die Kommunikation mit dem Server unabhängig voneinander [Gar05].

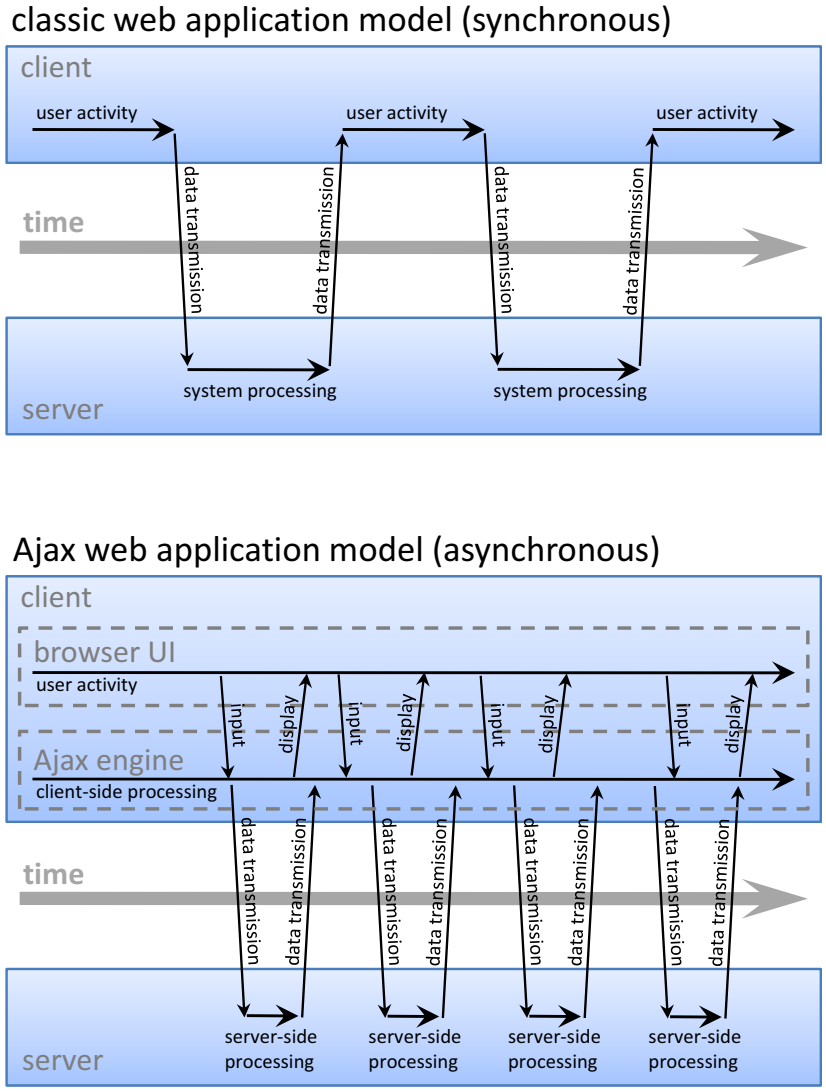


Abbildung 2.4: Vergleich des traditionellen synchronen Interaktionsmusters bei Web-Anwendungen und dem asynchronen Ajax-Muster

In Abbildung 2.4 ist der Vergleich zwischen der dem traditionellen synchronen Interaktionsmuster bei Web-Anwendungen und dem asynchronen Ajax-Muster zu sehen [Gar05]. Eine Benutzeraktion, die bei der traditionellen Interaktion einen HTTP-Request zur Folge hat, nimmt beim Ajax-Interaktionsmuster die Form eines JavaScript-Aufrufes an die Ajax-Engine an. Sofern eine Aktion des Benutzers keine Server-Antwort erfordert, wie z.B. Datenvalidierung oder einfache Navigation, wird diese von der Engine selbst bearbeitet. Wenn die Ajax-Engine für eine Antwort etwas vom Server benötigt (z.B. für die Weiterverarbeitung von Daten oder das Abfragen von aktualisierten Daten), geschieht dies asynchron mittels XML. Durch dieses Vorgehen wird die Interaktion zwischen dem Benutzer und der Anwendung nicht gestört [Gar05].

2.3.1 Google Web Toolkit

Das Google Web Toolkit (GWT) ein Java-Entwicklungs-Framework und soll die Arbeit beim Erstellen komplexer Web-Anwendungen und insbesondere Ajax-Anwendungen erleichtern. Durch den „Java-nach-JavaScript-Compiler“ ist es möglich Ajax-Anwendungen sowohl client- als auch serverseitig in der Programmiersprache Java zu entwickeln. Dieser Compiler übersetzt den Java-Code in Browserkonformes JavaScript und HTML [Goo07].

2.3.1.1 Vorteile

Durch die Verwendung des GWT ergeben sich einige Vorteile für die Entwicklung von Webanwendungen: [Goo07]

- **Programmiersprache:** Die gesamte Anwendung kann in der Programmiersprache Java entwickelt werden. Hierbei kommen die konzeptionellen Vorteile von Java im Gegensatz zu JavaScript, wie z.B. strikte Objektorientierung und ausgefeiltes Exception-Handling, zum Tragen.
- **Entwicklungsumgebung:** Die Entwicklung kann in einer bewährten Entwicklungsumgebung stattfinden, wodurch sich Vorteile wie etwa Code-Vervollständigung und Typüberprüfungen ergeben.[13]
- **Debugging:** Das GWT bietet vollständiges Round-Trip-Debugging. Client- und Server-Code können in einer IDE debuggt werden.
- **Testen:** Durch die Integration von JUnit ist es möglich automatisierte Tests durchzuführen.
- **Oberfläche:** Bei der Entwicklung der Benutzeroberfläche ist es möglich, dynamische, wieder verwendbare Komponenten (Widgets) zu verwenden. Zusätzlich zu den enthaltenen Standard Widgets können Widgets selbst entwickelt werden. Außerdem können Widgets-Libraries von anderen Entwicklern eingebunden werden.[14]
- **Browser-Kompatibilität:** Der Compiler wandelt den Java Code in HTML und JavaScript um, so dass sich der Entwickler nicht um die Kompatibilitätsprobleme mit verschiedenen Browsern kümmern muss.

2.3.1.2 RPC

Das GWT bietet eine Unterstützung für die *Remote Procedure Calls (RPC)*. Ein RPC ist ein entfernter Aufruf einer Prozedur und ist ein Hauptbestandteil einer Ajax-Anwendung (siehe Abschnitt 2.3).

[13] In dieser Arbeit wurde hierfür Eclipse verwendet

[14] In dieser Arbeit wurde unter anderem die „GWT Widet Libray“ verwendet. Siehe hierzu Kapitel 5.2.

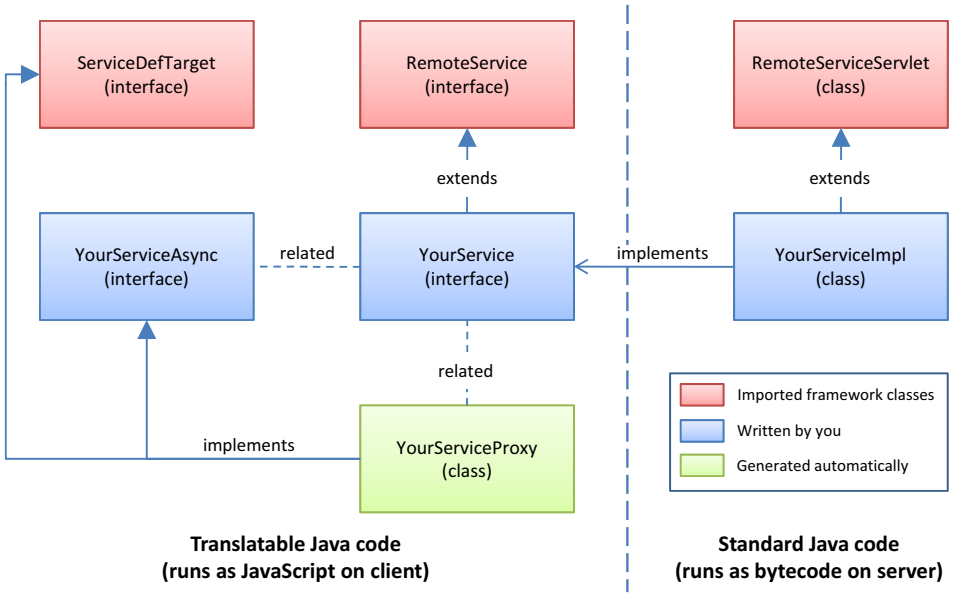


Abbildung 2.5: GWT RPC Diagramm

Abbildung 2.5 zeigt den GWT RPC Mechanismus [Goo07]. Mit dieser Programmierschnittstelle muss sich der Entwickler nicht mehr um die Details der asynchronen Kommunikation mittels HTTPRequest oder das Serialisieren von Objekten kümmern. Clientseitig müssen lediglich ein synchrone und eine asynchrone Schnittstelle mit den vom Service angebotenen Operationen definiert werden.

Zuerst muss ein Java Interface definiert werden, dass das GWT `RemoteService` Interface erweitert:

```
public interface UserService extends RemoteService {
  public String registerUser(User user);
}
```

Das zweite Interface für den Client entspricht dieser synchronen Schnittstelle `MyService` und wird benutzt um die Service Methode asynchron aufzurufen. Hierbei ist der Klassenname per Namenskonvention um das Suffix „Async“ zu erweitern. Die Operationen müssen den Rückgabewert void und einen Callback als zusätzlichen Parameter haben.

```
interface UserServiceAsync {
  public void registerUser(User user, AsyncCallback callback);
}
```

Serverseitig wird eine Klasse implementiert, die das `RemoteServiceServlet` erweitert und das Interface `UserService` implementiert.[15] Hierbei handelt es sich um das Servlet, das sich um die Nachrichtenverarbeitung und das Serialiseren der übertragenen Objekte kümmert.

[15] Das RemoteServiceServlet ist eine Klasse die von der Standard Java Klasse HttpServlet abgeleitet ist.

```
public class UserServiceImpl extends RemoteServiceServlet
                                              implements UserService {

  public UserRegistration registerUser(User user){
    ...
    return userRegistration;
  }
}
```

Um den Service aufzurufen wird zuerst ein Service-Proxy mittels `GWT.create()` erzeugt. Das Generieren der Proxy Klasse erfolgt automatisch im Hintergrund. Nachdem der Server-Proxy initialisiert ist, wird ein Callback-Objekt erzeugt mit dem der Service aufgerufen wird.

```
public void registerUser(User newUser) {
  UserServiceAsync userService = (UserServiceAsync)
                                      GWT.create(UserService.class);
  ServiceDefTarget target = (ServiceDefTarget) userService;
  target.setServiceEntryPoint(GWT.getModuleBaseURL() + SERVICE_URI);

  userService.registerUser(newUser, new AsyncCallback() {
    public void onSuccess(Object retValue) {
      ...
    }
    public void onFailure(Throwable error) {
      ...
    }
  });
}
```

Wenn der Aufruf des Services fehlerfrei war, wird vom Framework `onSuccess()` mit dem Rückgabeobjekt des Services aufgerufen. Im Fehlerfall, wenn eine Exception geworfen wird, wird `onFailure()` aufgerufen.

2.3.1.3 Besonderheiten

Das GWT bietet dem Entwickler zwei Modi zur Entwicklung an. Den *Hosted Mode* und den *Web Mode*. Im *Hosted Mode* wird die Anwendung in einer virtuellen Java-Maschine ausgeführt und lässt sich mit einer beliebigen Java-Entwicklungsumgebung debuggen. In diesem Mode wird der Client-Code nicht in JavaScript umgewandelt und die Darstellung wird in einem von Google mitgelieferten Web-Browser ausgeführt. Im *Web Mode* ausgeführt wird der Java Code vom Compiler nach JavaScript umgewandelt. Die Anwendung kann nun in einem Web-Browser ausgeführt werden.

In der derzeitigen GWT Version (1.4.59) werden vom Compiler nur Sprachkonstrukte der JSE2 Version 1.4 unterstützt.

2.4 Privacy

In Abschnitt 2.4.1 wird erläutert, was unter dem Begriff Privacy zu verstehen ist. Abschnitt 2.4.2 gibt einen Einblick in die Grundprinzipien des Datenschutzes. Abschnitt 2.4.3 geht auf die Personalisierung von Inhalten ein. Abschließend diskutiert Abschnitt 2.4.4 das Vertrauen in Anwendungen.

2.4.1 Begriff

Im Deutschen wird der Begriff „Privacy“ oftmals mit „Privatsphäre“ oder „Datenschutz“ übersetzt. Gemeint ist der Schutz der Menschen vor der Verarbeitung persönlicher Daten [Gar03].

Laut Alan F. Westin [Wes67] ist es jedoch nicht möglich, eine abschließende Definition von Privacy zu finden:

„no definition of privacy is possible, because privacy issues are fundamentally matters of values, interests, and power“

Ein in der Literatur oft zitierter Definitionsversuch von ihm lautet:

„Privacy is the claim of individuals, groups or institutions to determine for themselves when, how, and to what extent information about them is communicated to others.“

Aus dieser Definition geht hervor, dass Personen einen Anspruch darauf haben, selbst über die Weitergabe persönlicher Informationen zu bestimmen. Dies wird auch im deutschen Recht durch das BVerfG-Volkszählungsurteil von 1983 [Bun83] bestärkt. Hier wurde erstmals anerkannt, dass es ein Grundrecht auf informationelle Selbstbestimmung gibt.

„Das Grundrecht gewährleistet insoweit die Befugnis des Einzelnen, grundsätzlich selbst über die Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen.“

2.4.2 Grundprinzipien

Die Grundlage für alle modernen Datenschutzgesetze bilden die *„United Nations guidelines concerning Computerized personal data files“*, die von den Vereinten Nationen verfasst worden ist [Com90]. In diesen werden unter anderen folgende Grundsätze definiert:

- **Principle of lauwfulness and fairness:** Es sollen keine Informationen über Personen in unehrlicher oder rechtswidriger Weise erhoben oder verarbeitet werden.
- **Principle of accuracy:** Personen, die Daten erheben, sind dazu verpflichtet, die Daten regelmäßig zu kontrollieren und auf Richtigkeit und Aktualität zu überprüfen.
- **Principle of the purpose- specification:** Die erhobenen Daten dürfen nur zu dem Zweck verwendet werden, zu dem sie erhoben wurden.

- **Principle of interested-person access:** Personen haben das Recht zu entscheiden und zu erfahren, welche Daten über sie erhoben werden.
- **Principle of non-discrimination:** Daten, die zu einer Diskriminierung der betroffenen Person führen können, dürfen nur unter speziellen Voraussetzungen erhoben werden.

Diese Richtlinien legen einen Mindeststandard fest, der bei der nationalen Gesetzgebung berücksichtigt werden sollte.

2.4.3 Personalisierung

Aus aktuellen Benutzerstudien [Inc07] geht hervor, dass Online-Nutzer personalisierte Inhalte zu schätzen wissen, aber sehr über ihre Privatsphäre besorgt sind.

Beispiele für die Bedenken sind:

- Personen, die online in einem Buchladen einkaufen, bekommen Empfehlungen über Bücher, sind aber besorgt, ob die von ihnen getätigten Käufe wirklich geheim gehalten werden
- Personen, die im Internet etwas suchen, freuen sich, dass die Suche durch die Personalisierung verbessert wird, sind aber beunruhigt, dass alle vorherigen Suchanfragen gespeichert werden.
- Schüler erfreuen sich an personalisierten Lernsystemen, fragen sich aber, ob auch andere Personen Zugriff auf das eigene Wissensniveau haben.

Die Qualität der Personalisierung hängt von der Menge der Daten des Benutzers ab. Eine Erhöhung der Personalisierung zieht aber eine Einschränkung der Privatsphäre nach sich (und umgekehrt) [Kob07]. Abbildung 2.6 soll diese Beziehung visualisieren.

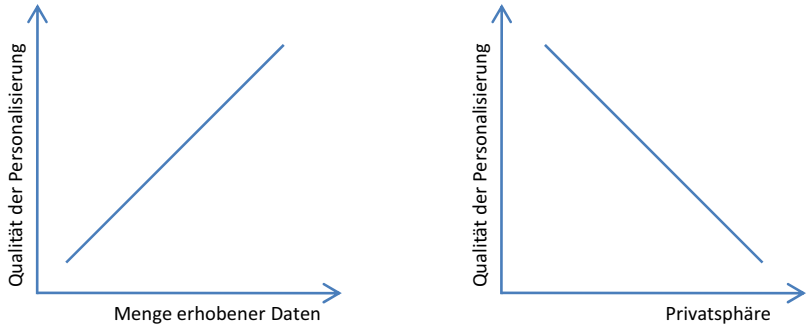


Abbildung 2.6: Beziehung zwischen Qualität der Personalisierung und Privatsphäre

Aktuelle Forschungen haben ergeben, dass für die Akzeptanz von personalisierten Systemen mehr als nur der Grad von Privatsphäre und Personalisierung berücksichtigt werden muss. Das Gebiet „privacy-enhanced personalization“ [KCS06, KC05a] versucht, die Ziele und Methoden

der Benutzermodellierung und Personalisierung mit Privatsphäre-Überlegungen in Einklang zu bringen um eine bestmögliche Personalisierung innerhalb der Grenzen der Privatsphäre zu erreichen [Kob07].

2.4.3.1 Privatsphäre Kalkül

Privacy Theorien besagen, dass das Verhalten von Personen das Ergebnis einer situations spezifischen Kosten-Nutzen-Analyse ist. In dieser werden potentielle Risiken der persönlichen Information gegen den persönlichen Vorteil abgewogen [Kob07].

Oftmals fehlt es den Internet-Benutzern an ausreichend Informationen um angemessene Privatsphäreentscheidungen zu treffen. Zum Beispiel unterschätzen Personen die Möglichkeit, dass sie aufgrund der Veröffentlichung bestimmter Daten identifiziert werden können, oder sie sind nicht mit den Datenschutz-Erklärungen der Internetseite vertraut. Acquisti [Acq04] diskutiert die Möglichkeit von übertriebener Vernachlässigung rationaler Privatsphäreentscheidungen. Durch diese kann es zu einer Überschätzung eines kleinen aber sofortigen Nutzens und zu einer Unterschätzung der negativen Auswirkungen in der Zukunft kommen.

Die Faktoren, die eine Rolle beim „privacy calculus“ eine Rolle spielen, umfassen persönliche und kulturelle Gesichtspunkte, die Informationsart, den Empfänger der Information, die Höhe des Nutzens und der Kontrolle sowie Vertrauensgesichtspunkte.

2.4.3.2 Gruppen

In den Privatsphäre-Studien hat Alan F. Westin [WA91] drei Gruppen von Verbrauchern unterschieden, die verschiedene Stufen der Privatsphäre-Besorgnis repräsentieren:

- **Privacy fundamentalists:** Privatsphäre-Fundamentalisten sind extrem besorgt, wenn es um ihre persönlichen Daten geht. Sie möchten ihre Daten sogar dann nicht preisgeben, wenn Sicherheitsmechanismen vorhanden sind.
- **Privacy unconcerned:** Privatsphäre unbesorgte Personen haben wenig Bedenken um ihre Privatsphäre. Sie sind nicht ängstlich, inwieweit andere Personen oder Organisationen ihre persönlichen Informationen nutzen.
- **Privacy pragmatists:** Privatsphäre-Pragmatiker sind zwar um ihre persönlichen Daten besorgt, aber nicht so stark wie die Fundamentalisten. Sie sind eher dazu geneigt, persönliche Informationen preis zu geben, wenn sie z.B. den Grund dafür verstehen, einen Vorteil sehen oder sehen, dass Sicherheitsmechanismen vorhanden sind.

Das Verhältnis der Gruppen ist 1:1:2 und wird in Abbildung 2.7 visualisiert. Das genaue Verhältnis variiert in den Umfragen, wobei die Zahl der Pragmatiker in den letzten zwei Dekaden gestiegen und die Zahl der unbesorgten Personen gesunken ist [KC05b].

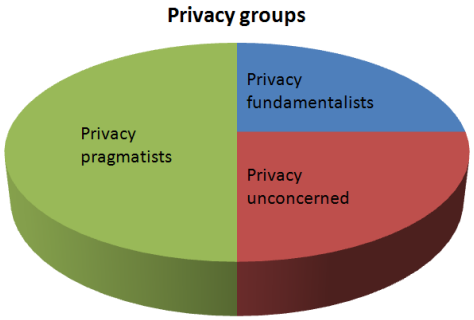


Abbildung 2.7: Übersicht über Privatsphäre-Gruppen

2.4.3.3 Informationsart und -nutzen

Aus den Umfragen geht hervor, dass die Art der preisgegebenen Information eine entscheidende Rolle bei dem Verhalten der Benutzer spielt [BKN07].

Demnach sind Benutzer gerne bereit Informationen über generelle demografische Daten, den Lebensstil, den persönlichen Geschmack oder Hobbys preiszugeben. Sie sind weniger bereit, Informationen über ihr Internetverhalten, Einkäufe oder erweiterte demografische Daten anzugeben. Am wenigsten sind sie dazu bereit, finanzielle Informationen, Kontaktinformationen, die Kreditkartennummer und die Sozialversicherungsnummer preiszugeben.

Hierbei wird der Wert der Personalisierung berücksichtigt, also welchen persönlichen Nutzen der Anwender in den angebotenen Diensten sieht.

Es wurden sieben verschiedene Arten von Motiven identifiziert, die sich in „Äußerliche“ und „Innerliche“ Vorteile des Nutzers gruppieren lassen: [HTG06]

- **Äußerliche Vorteile**
  - *Geldersparnis (monetary savings)*: Rabatte, Gutscheine, Geschenke, etc.
  - *Zeitersparnis (time savings)*: Bessere Effizienz und Komfort.
  - *Selbstdarstellung (self-enhancement)*: Aufwertung der eigenen Person gegenüber anderen.
  - *Soziale Einstellung (social adjustment)*: Einordnung der eigenen Person zur Integration in die gewünschte soziale Gruppe.
- **Innerliche Vorteile**
  - *Vergnügen (pleasure)*: Der Umgang soll Spaß machen und nicht frustrieren.
  - *Neuheit (novelty)*: Unbekanntes erkunden und Wissensbedarf befriedigen.
  - *Uneigennützigkeit (alturism)*: Anderen ohne egoistischen Hintergrund helfen.

Wenn Benutzer den persönlichen Nutzen erkennen, sind sie eher gewillt die erforderlichen Informationen anzugeben. Daher sollten die Vorzüge der erhobenen Daten von den Anbietern gut kommuniziert werden.

2.4.4 Vertrauen

Das Vertrauen in eine Webseite ist ein wichtiger Faktor, um den Benutzer zu motivieren persönliche Daten preiszugeben. Wenn Benutzer den Anbietern eines Dienstes nicht vertrauen und nicht wissen, wie ihre persönlichen Informationen verwendet werden, lehnen sie oftmals die Angabe ab oder geben absichtlich falsche Angaben an [Kob07].

Es gibt eine Reihe von Faktoren, die auf das Vertrauen Einfluss nehmen. Unter ihnen sind positive Erfahrungen in der Vergangenheit, Design und Verhalten einer Seite, Reputationen der Webseiten-Eigentümer, Vorhandensein von Datenschutz-Erklärungen und Vorhandensein von Datenschutz-Zertifikaten.

Positive Erfahrungen in der Vergangenheit

Positive Erfahrungen mit einer Web-Seite gehören besonders bei Langzeit-Verbindungen zu den Vertrauensfaktoren. Laut Kobsa [Kob07] sollte nicht sofort nach allen Informationen des Benutzers gefragt werden. Den Benutzern sollte die Möglichkeit gegeben werden, die persönlichen Informationen im Laufe der Zeit zu ergänzen, wenn schon ein Vertrauensverhältnis besteht.

Design und Verhalten einer Seite

Verschiedene Interface-Design Elemente und operationale Charakteristika sollen das Vertrauen der Benutzer stärken: [Fog03]

- Geringe Error-Quote
- Professionelles Design und Usability
- Vorhandensein von Kontaktinformationen
- Verweise von einer glaubhaften/namhaften Website
- Verweise zu weitergehenden Materialien
- Updates seit dem letzten Besuch
- Schnelle Antwort auf Kundenfragen
- E-Mail Bestätigungen für alle Transaktionen

Reputationen der Webseiten-Eigentümer

Laut Brusilovsky [BKN07] ist die Reputation des Webseiten-Eigentümers ein weiter Vertrauensfaktor. Wenn die Webseite bekannt ist und häufig besucht wird, sind die Benutzer eher geneigt identifizierende Informationen anzugeben.

Vorhandensein von Datenschutz-Erklärungen

Datenschutz-Erklärungen (privacy statements oder privacy policies) beschreiben den Umgang mit persönlichen Daten. Der Effekt auf das Vertrauen der Benutzer konnte bislang nicht geklärt werden. In einigen Studien hatte das reine Vorhandensein von privacy statements einen positiven Effekt auf das Vertrauen und die Preisgabe von Daten. Die Webserver-Logs zeigten aber, dass weniger als ein Prozent der Besucher auf diese zugreifen [BKN07].

Vorhandensein von Datenschutz-Zertifikaten

Das reine Vorhandensein von Zertifikaten hat einen klaren, positiven Einfluss auf das Vertrauen der Benutzer und auf das Preisgeben ihrer privaten Daten. Einige Forschungen belegen jedoch, dass einige Zertifizierungsorganisationen keine gute Prüfung der Zertifikatinhaber betreiben. Aus Studien geht hervor, dass Seiten mit Zertifikat ein größeres, privatsphäreangreifendes Geschäftsgebaren haben als Seiten ohne Zertifikat [Ede06].

3 Systemanalyse

In diesem Kapitel werden die Anforderungen an das in dieser Arbeit entwickelte System vorgestellt. Hierzu wird in Abschnitt 3.1 zunächst das Ausgangsszenario vorgestellt. In Abschnitt 3.2 werden Projekte, die an die Themenbereiche dieser Arbeit angrenzen, vorgestellt und miteinander verglichen. Abschnitt 3.3 stellt die Vorgehensweise der Anforderungsanalyse vor. Anschließend werden in Abschnitt 3.4 die funktionalen Anforderungen beschrieben. Hierzu werden die Anwendungsfälle in Abschnitt 3.4.1 dargestellt und entsprechenden Systemabläufe in Abschnitt 3.4.2 aufgezeigt. Abschnitt 3.4 stellt die nichtfunktionalen Anforderungen an das System dar.

3.1 Ausgangsszenario

Es gibt viele Wege Informationen zu veröffentlichen und Informationen zu erhalten. Innerhalb von Firmen und auch an der HAW (Hochschule für Angewandte Wissenschaften Hamburg) werden Informationen, die keinen bestimmten Empfänger haben, meist an einem Schwarzen Brett veröffentlicht oder über E-Mail-Verteilerlisten versendet. Weitere Informationen werden auf speziellen Internet-Seiten veröffentlicht.

Aufgrund der Menge der veröffentlichten Informationen ist es für eine Person oftmals nicht einfach die relevanten Informationen zu finden und einen Überblick zu behalten. Hinzu kommt, dass die gefundenen Inhalte unterschiedlich wahrgenommen werden [SC05].

Die suchende Person ist mit der Suche und der Organisation von Inhalten auf sich alleine gestellt. Die Informationen müssen eigenständig gesichtet und nach persönlicher Relevanz bewertet werden. Zudem muss sich die Person in die richtigen E-Mail-Verteilerlisten eintragen bzw. müssen diese Listen gepflegt werden. Dies kostet Zeit und oftmals werden Informationen übersehen oder der Inhalt von Informationen ist nicht für jeden Empfänger gleichermaßen interessant.

Beispielsweise wird eine Mitteilung, die besagt, dass ein System nicht zur Verfügung steht, am Schwarzen Brett veröffentlicht und per Verteilerliste an die gesamte Firma versendet. Der Aushang am Schwarzen Brett könnte von einigen betroffenen Personen übersehen werden oder von anderen Anzeigen überdeckt sein. In der Verteilerliste können sich auch Personen befinden, die nicht mit dem System arbeiten und keine Berührungspunkte mit diesem haben.

Um einen Benutzer bei der Informationsrecherche zu unterstützen und eine benutzerfreundliche Informationsbeschaffung zu ermöglichen, soll ein Benutzerprofil mit ontologiebasierten

Interessen helfen. Dieses Vorwissen über eine Person kann genutzt werden um die persönlich relevanten Informationen einzugrenzen. Die zur Verfügung stehenden Informationen können bewertet und einer Person empfohlen werden. Dies erleichtert die Suche für die Person und ermöglicht eine effiziente Übersicht.

Der Benutzer kann mit Hilfe eines solchen Systems eine gewichtete Übersicht bekommen und diese mit seinen Profileigenschaften eingrenzen. Zudem kann er bei der Veröffentlichung von Mitteilungen unterstützt werden, indem er eine automatische Rückmeldung und Bearbeitungsmöglichkeit der auf die Mitteilung zutreffenden Gebiete erhält.

3.2 Verwandte Arbeiten

Nach Kenntnis des Autors existieren derzeit keine Projekte, die sich mit sowohl mit ontologiebasierten Interessenprofilen als auch mit offenen Profilen beschäftigen.

Es gibt Projekte, die sich mit der Filterung von Nachrichten beschäftigen. Unter diese Projekte fallen Google Alerts (BETA) und PressWatch. Ein weiterer angrenzender Bereich ist die digitale Pinnwand, an der Mitteilungen veröffentlicht und verwaltet werden können. Hierzu wird auf das Projekt Stickees eingegangen.

Mit **Google Alerts (BETA)** [Goo08] kann sich der Benutzer per E-Mail über selbst gewählte Suchbegriffe informieren lassen. Derzeit kann der Benutzer aus vier vorgegebenen Typen von „Alerts“ auswählen: News, Web, News&Web und Groups. Diese repräsentieren die verschiedenen Bereiche von Google. Bei News wird z.B. die Google Nachrichtenseite nach dem gewählten Suchbegriff durchsucht. Wenn sich ein neuer Artikel unter den ersten zehn Ergebnissen befindet wird der Benutzer informiert. Die Zielgruppe dieses Dienstes sind Internetbenutzer, die benachrichtigt werden wollen und nicht selbst in den verschiedenen Bereichen suchen möchten.

PressWatch [Pre08] beobachtet über 120.000 Zeitschriften und Zeitungen aus über 100 Ländern und über 5.000 Internet-Portale weltweit. Damit erhalten Benutzer alle Neuigkeiten, die ihr Business beeinflussen. Unter den Angeboten sind PrintWatch (Beobachtung von Printmedien), WebWatch (Beobachtung von Online-Medien), OpinionWatch (Beobachtung von Weblogs, Newsgroups und Foren nach Kundenauswahl) und TV-Watch (Beobachtung von Nachrichten- und Magazinformaten). Die Zielgruppe dieses Dienstes sind Firmen und Business-Kunden.

Das Projekt **Stickees** [Ste06] erfüllt die klassische Pinnwand-Metapher und bietet die Möglichkeit die Übersicht über digitale Daten zu behalten. Diese Daten können in inhaltliche und zeitliche Ordnung gebracht werden. Mit dieser Lösung können z.B. alle Materialien zu einem Projekt aufbewahrt und organisiert werden. Die Zielgruppe dieses Dienstes sind sowohl Privatpersonen als auch Firmen.

Abgrenzung

Abbildung 3.1 zeigt eine Übersicht über einige Möglichkeiten der einzelnen Projekte.

-: nicht vorhanden +: vorhanden 0: keine Aussage	Eigene Suchbegriffe	Gewichtung der Begriffe	Ontologien/ Kategorien	Kostenlose Nutzung	Übersicht über Gesamtmenge der Beiträge	Veröffentlichen eigener Beiträge	Community
Profil.Info.Sys	-	+	+	+	+	+	+
Google Alerts	+	-	-	+	-	-	-
PressWatch	+	+	-	-	o	-	-
Stickees	-	-	-	+	+	+	+

Abbildung 3.1: Abgrenzung verwandter Projekte

Bei Google Alerts können die Suchbegriffe frei gewählt, aber nicht nach Priorität gewichtet werden. Es ist ebenfalls nicht möglich angrenzende oder verwandte Wörter automatisch mit in die Suche einzubeziehen, da keine Ontologien oder Kategorien verwendet werden.

PressWatch verwendet ebenfalls keine Ontologien, ist aber eine Dienstleistung, die auch angrenzende Bereiche umfasst und sogar Printmedien durchsucht. Es ist ein kommerzieller Service, der mit monatlichen Kosten verbunden ist.

Stickees erfüllt die Pinnwand-Metapher und schafft einen Überblick über die veröffentlichten Mitteilungen und Daten. Derzeit ist das System aber noch nicht personalisierbar bzw. es unterstützt keine Benutzerprofile. Eine Benachrichtigungsfunktion ist nicht integriert und somit eignet sich Stickees eher zur Visualisierung und Verwaltung von Materialien und Notizen.

3.3 Vorgehensweise

Dieses Kapitel beschäftigt sich mit der Vorgehensweise der Anforderungsanalyse. Zunächst werden in Abschnitt 3.3.1 die Faktoren, die einen Einfluss auf die Auswahl der Ermittlungstechnik haben, vorgestellt. Die in dieser Arbeit verwendete Ermittlungstechnik wird erläutert und der Kreis der befragten Personen in Abschnitt 3.3.2 diskutiert. Abschließend stellt Abschnitt 3.3.3 den Aufbau und die verfolgten Ziele der gewählten Ermittlungstechnik dar.

3.3.1 Ermittlungstechnik

Zur Ermittlung von Wissen wurden eine Vielzahl von Techniken entwickelt, die sich für den Einsatz bei unterschiedlichen Randbedingungen eignen. Die richtige Technik für alle Anforderungen in einem Projekt gibt es jedoch nicht [RG02].

Rupp gliederte die Einflussfaktoren für die Anforderungsermittlung in drei Kategorien: **Mensch**, **organisatorische Rahmenbedingungen** und **fachlicher Inhalt**. Diese Faktoren haben direkte Auswirkungen auf die Anwendung einer Ermittlungstechnik.

Jede dieser Kategorien hat mehrere konkrete Einflussfaktoren, die unterschiedlich stark ausgeprägt sind und unterschiedliche Relevanz für Projekte haben.

3.3.1.1 Menschliche Einflussfaktoren

Menschen haben einen hohen Einfluss auf die Anforderungsermittlung, da in dieser Phase eines Projektes funktionierende Kommunikation notwendig ist. Folgende konkrete Faktoren werden unterschieden:

- **Motivation:** Motivation der Stakeholder, aktiv am Anforderungsanalyseprozess mitzuwirken.
- **Kommunikative Fähigkeiten:** Stakeholder können eigene Gedanken korrekt artikulieren und fremde treffend interpretieren.
- **Art des Wissens:** Die geforderten Informationen sind den Stakeholdern bewusst oder die zu ermittelnden Anforderungen sind implizites Wissen, das im Unbewussten verborgen ist.
- **Machtsituation und Gruppendynamik:** Problematische Machtsituation zwischen den Projektbeteiligten.
- **Abstraktionsvermögen:** Stakeholder können die Beschreibung ihrer Wünsche auf den fachlichen Kern beschränken.
- **Homogenität der Stakeholdermeinungen:** Viele Stakeholder mit sehr unterschiedlichen, sich widersprechenden Forderungen an das System.

3.3.1.2 Organisatorische Rahmenbedingungen

Organisatorische Gegebenheiten eines Projektes beeinflussen ebenfalls die Ermittlung von Anforderungen. Folgende Faktoren zählen zu diesen Bedingungen:

- **Neuentwicklung oder Altsystemerweiterung:** Es muss berücksichtigt werden, in wieweit das zu entwickelnde (Teil-)System in ein bestehendes System integriert werden soll. Bei einer Neuentwicklung muss keine Rücksicht genommen werden.
- **Komplexität des Marktes:** Sofern keine Individualentwicklung für einen Kunden gemacht wird, müssen Markteinflüsse beachtet werden.
- **Vertragsmodell:** Juristische und vertragliche Randbedingungen wie z.B. Festpreisprojekt, Aufraggeber, Ausschreibungsart.
- **Projektbudget:** Flexibilität der Projektdauer und Projektmittel.

- **Räumliche Verteilung der Stakeholder:** Räumliche Entfernung der Stakeholder z.B. weltweit verteilt.
- **Zeitliche Verfügbarkeit der Stakeholder:** Engpässe der eigentlichen Arbeit der Stakeholder und Kostenaufwand durch Zeitverlust.
- **Anzahl der Stakeholder:** Bei steigender Zahl der Stakeholder erhöht sich der Ermittlungs- bzw. Analyseaufwand.

3.3.1.3 Fachlicher Inhalt der Anforderungen

Verschiedenen inhaltlichen Ausprägungen der zu ermittelnden Anforderungen können Einfluss auf die Wahl der Ermittlungstechnik haben. Folgende Ausprägungen werden unterschieden:

- **Kritikalität des Systems:** Wenn ein Fehlverhalten des Systems kritisch ist und hohe Kosten verursacht oder im schlimmsten Fall Menschenleben gefährdet sind.
- **Systemumfang:** Die Anzahl der Anforderungen spielt als Maß für den Umfang des Systems eine Rolle.
- **Komplexität der Systemabläufe:** Komplexe Systeme sind schwerer zu erfassen und zu beschreiben.
- **Beobachtbarkeit der Arbeitsschritte:** Die Stakeholder müssen die Arbeitsschritte selbst beschreiben, wenn diese nur schwer zu beobachten sind.
- **Art der Anforderungen:** Funktionale Anforderungen sind bei der Ermittlung weniger kritisch, da sie Arbeitsabläufen entsprechen und daher leicht vorstellbar sind.
- **Erfahrung im Fachgebiet:** Es können weniger Vorschläge und Erfahrungen eingebracht werden, wenn dem Analytiker das Fachgebiet unbekannt ist.

Laut Rupp ist es bei bei der Analyse der geeignetsten Ermittlungstechnik für ein Projekt ausreichend, sich auf die drei bis sechs am stärksten ausgeprägten Faktoren zu beschränken [RG02].

Als stärkste Risikofaktoren werden in diesem Teil der Arbeit die organisatorischen Randbedingungen **Neuentwicklung**, **Fixiertes, knappes Projektbudget** und **hohe Zahl von Stakeholdern** angesehen. Weiterhin werden der menschlichen Einflussfaktor **Divergierende Stakeholdermeinungen** und der fachliche Inhalt **Geringe Beobachtbarkeit** als starke Einflussfaktoren gewertet.

Es handelt sich bei dieser Arbeit um ein Forschungsprojekt und somit um eine Neuentwicklung bei der keinerlei Rücksicht auf bestehende Systeme genommen werden muss. Das fixierte, knappe Projektbudget ist durch die Prüfungsordnung vorgegeben, die eine zeitliche Beschränkung der Arbeit vorsieht. Im Szenario der HAW (siehe Abschnitt 3.1) gibt es verschiedene Benutzer-Rollen und eine hohe Zahl von potentiellen Benutzern. Durch die unterschiedlichen

Tätigkeitsgebiete der Benutzer/Stakeholder ist die Wahrscheinlichkeit für divergierende Meinungen sehr groß. Hinzu kommt, dass es sich bei diesem Szenario um eine freiwillig durchgeführte Tätigkeit (Informieren über Informationen) handelt. Die beteiligten Personen werden sich vermehrt in ihrer freien Zeit informieren, was die Beobachtbarkeit der Anforderung erschwert.

Abbildung 3.2 zeigt den Einfluss der vorgestellten Risikofaktoren auf die Anwendbarkeit von Ermittlungstechniken [RG02].

- - : gar nicht geeignet
- : nicht gut geeignet
0 : Kein Einfluss – geeignet
+: gut geeignet
++ : sehr gut geeignet

	Kreativität			Beobachtung		Befragung			
	Brainstorming	Wechsel d. Perspektive	Mind Mapping	Feldbeobachtung	Apprenticing	Fragebogen	Interview	Selbstaufschreibung	On-Site-Customer
Mensch									
Geringe Motivation	-	-	-	+	-	+	+	- -	- -
Schlechte kommunikative Fähigkeiten	-	-	0	-	++	0	-	--	-
Implizites Wissen	+	+	+	++	++	- -	-	- -	-
Geringes Abstraktionsvermögen	-	-	-	++	++	+	+	-	+
Divergierende Stakeholdermeinungen	-	-	-	-	-	++	0	-	- -
Problematische Gruppendynamik	- -	0	-	-	++	0	0	+	+
Organisatorische Rahmenbedingungen									
Neuentwicklung	++	++	++	0	0	+	+	+	+
Altsystemerweiterung	0	0	0	+	+	+	+	+	+
Komplexer Markt	++	++	+	+	+	+	-	- -	+
Individualentwicklung	0	0	0	+	+	+	+	++	++
Produktentwicklung	++	++	++	-	- -	+	-	-	- -
Fixiertes, knappes Projektbudget	0	0	0	-	- -	+	+	+	-
Hohe Verteilung der Stakeholder	- -	-	-	0	0	+	-	++	-
Schlechte Verfügbarkeit der Stakeholder	-	-	-	++	- -	+	-	- -	- -
Hohe Zahl von Stakeholdern	-	0	0	-	-	+	-	- -	-
Fachlicher Inhalt der Anforderungen									
Hohe Kritikalität des Systems	0	+	0	+	- -	+	+	+	++
Großer Systemumfang	0	0	0	++	+	- -	+	+	+
Hohe Komplexität der Systemabläufe	+	+	++	-	+	- -	+	+	+
Geringe Beobachtbarkeit	+	+	+	- -	+	+	+	+	+
Nicht funktionale Anforderungen	-	+	0	-	+	- -	-	-	-
Unbekanntes Fachgebiet	0	0	0	+	++	- -	+	++	+

Abbildung 3.2: Einflussfaktoren auf die Anwendbarkeit von Ermittlungstechniken

Der Abbildung ist zu entnehmen, dass sich die Befragungstechnik **Fragebogen** sehr gut für diese Arbeit eignet. Der Fragebogen stellt einen effektive Technik dar, eine große Anzahl von Benutzern unter einem geringen Zeit- und Kostenaufwand in die Analyse mit einzubeziehen. Es besteht die Möglichkeit die Wünsche der Benutzer in offenen Fragen zu ermitteln. Desweiteren

ist die Auswertung relativ einfach und weitestgehend automatisierbar und somit bei fixiertem Zeitaufwand empfehlenswert [RG02].

Aus dem Fragebogen können die Benutzerprofile[1] ermittelt und die grundlegenden Anforderungen an das spätere System abgeleitet werden.

3.3.2 Befragte Personen

Um ein effektives Kosten-Nutzen-Verhältnis zu erhalten, wurde der Kreis der befragten Personen eingegrenzt und nicht jede einzelne in Frage kommende Person befragt.

Zunächst wurden hierfür die wichtigsten Gruppen der potentiellen Nutzer identifiziert. Am Beispiel der HAW sind dies:

- Professoren
- Fakultätsverwaltung
- Mitarbeiter
- Studenten

Aus jeder dieser Gruppen wurden stickprobenartig Personen ausgewählt. Diese bekamen einen Fragebogen per E-Mail geschickt. Der Fragebogen ist im Anhang A.2 zu finden.

Insgesamt wurden 34 Personen befragt. Davon 6 Professoren, 5 Personen aus der Fakultätsverwaltung, 6 Mitarbeiter und 17 Studenten.

3.3.3 Ziele

Der Fragebogen gliedert sich in drei Teile: **Allgemeine Angaben**, **Anwendungsspezifische Angaben** und **Sonstige Angaben**.

Im Abschnitt **Allgemeinen Angaben** werden Fragen zur Person gestellt. Es wird nach Alter, Haupttätigkeit an der Hochschule, Fachbereich und Erfahrung mit Internetanwendungen gefragt.

Diese Angaben dienen dazu die Benutzerkategorie zu bestimmen. Die Kategorie soll Aufschluss über eine eventuelle Rollenverteilung und Aufgaben/Anforderungsabgrenzung geben. Die Frage mit Bezug auf die Nutzung von Internetanwendungen zielt darauf ab, wie versiert die Benutzer im Umgang mit dieser Art von Anwendungen sind, da der entwickelte Prototyp ebenfalls eine Internetanwendung ist.

Der Abschnitt **Anwendungsspezifische Angaben** richtet sich an vier Informationsmöglichkeiten innerhalb der Hochschule:

[1] Siehe Grundlagenkapitel 2.1.1

- Ein Schwarzes Brett an dem Krankmeldungen, Veranstaltungspläne, GW-Kurse und ähnliches ausgehängt wird.
- Ein Schwarzes Brett an dem Gesuche oder Angebote zu Stellen, Wohnungen oder Sonstigem platziert werden können.
- E-Mail-Verteilerlisten über die an Informationen gelangt werden kann, bzw. über die Informationen an eine bestimmte Personengruppe gesendet werden können.
- Einen „News-Bereich“ auf der Hochschulhomepage über den Informationen bekannt gegeben werden können.

Zu jeder dieser Möglichkeiten werden spezifische Fragen zum Veröffentlichen und Empfangen von Informationen gestellt. Dadurch soll die Ist-Situation dokumentiert und neue Anforderungen entdeckt werden.

Der Abschnitt **Sonstige Angaben** deckt die nicht explizit hinterfragten Informationskanäle und deren Möglichkeiten ab.

3.4 Funktionale Anforderungen

Ausgehend von dem im Abschnitt 3.1 dargestellten Anwendungsszenario, sowie der Auswertung der versendeten Fragebögen, ergeben sich funktionale Anforderungen, die die Funktionalität der Software beschrieben.

3.4.1 Anwendungsfälle

Die nachfolgenden Anwendungsfälle sollen die funktionalen Anforderungen an das zu entwickelnde System verdeutlichen.

3.4.1.1 Mitteilung veröffentlichen

Dieser Anwendungsfall beschreibt das Veröffentlichen von Mitteilungen.

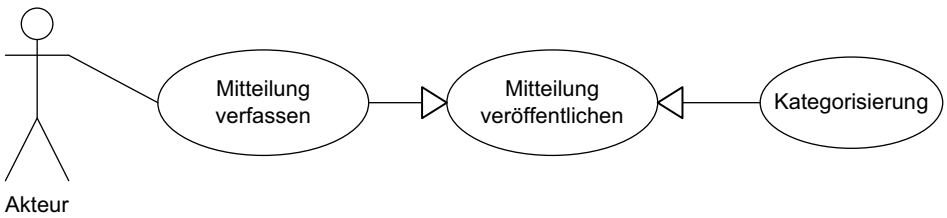


Abbildung 3.3: Anwendungsfalldiagramm Mitteilung veröffentlichen

Vor- und Nachbedingung: Der Akteur ist im System registriert und angemeldet. Am Ende des Anwendungsfalls ist eine neue Mitteilung im System verfügbar und der Benutzer erhält eine aktualisierte Liste seiner veröffentlichten Mitteilungen.

Beschreibung: Der Akteur benutzt die Internetanwendung und meldet sich durch Eingabe seines Nutzernamens und seines Passworts beim System an. Er erstellt eine neue Mitteilung und gibt hierfür die Gültigkeitsdauer, die Überschrift und den Inhalt der Mitteilung an. Die Mitteilung wird erstmals vorkategorisiert. Der Akteur erhält eine Rückmeldung über die automatisch zugeordneten Kategorien und hat die Möglichkeit diese zu bearbeiten. Anschließend wird die Mitteilung veröffentlicht. Die neue Mitteilung wird nun im System als aktuelle Mitteilung aufgenommen und der Akteur erhält eine aktualisierte Liste mit den von ihm veröffentlichten Mitteilungen.

3.4.1.2 Mitteilung erhalten

Dieser Anwendungsfall beschreibt den Erhalt von Mitteilungen.

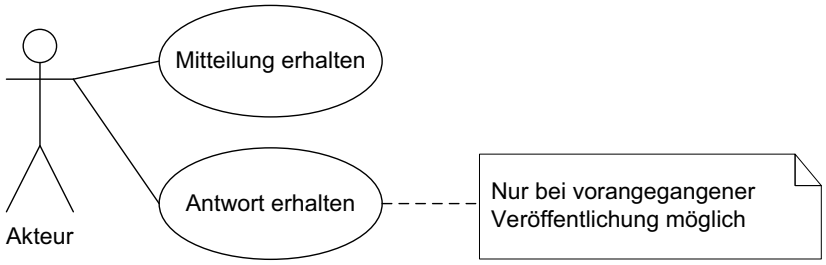


Abbildung 3.4: Anwendungsfalldiagramm Mitteilung erhalten

Vor- und Nachbedingung: Der Akteur ist im System registriert und angemeldet. Am Ende des Anwendungsfalls hat der Benutzer eine Mitteilung im Mitteilungs- bzw. Feedbackbereich der Anwendung.

Beschreibung: Dieser Anwendungsfall besteht aus zwei Teilen *Mitteilung erhalten* und *Antwort erhalten*. Bei beiden Teilen handelt es sich um Mitteilungen, die aber vom System unterschiedlich behandelt werden. In beiden dieser Fälle benutzt der Akteur die Internetanwendung und meldet sich durch Eingabe seines Nutzernamens und seines Passworts beim System an.

Mitteilung erhalten: Das System prüft, ob eine Übereinstimmung mit dem Profil des Benutzers und den aktuell veröffentlichten Mitteilungen vorliegt.[2] Wenn eine Übereinstimmung festgestellt wird, erscheint die entsprechende Mitteilung im Mitteilungsbereich des Benutzers.

Antwort erhalten: Sofern der Benutzer eine Mitteilung veröffentlicht hat, kann er Feedback zu dieser erhalten. Dies ist eine Art direkte Kommunikation zwischen zwei Benutzern, da die Antwort nicht veröffentlicht und somit nicht indiziert und kategorisiert wird. Unmittelbar nach dem

[2] Diese Überprüfung findet bei der Anmeldung und danach zeitgesteuert in einem bestimmten Intervall statt.

Versenden der Antwort (siehe 3.4.1.3) wird die Mitteilung dem Empfänger zugestellt und in dessen Feedbackbereich angezeigt.

3.4.1.3 Beantworten von Mitteilungen

Dieser Anwendungsfall beschreibt das Beantworten von Mitteilungen.

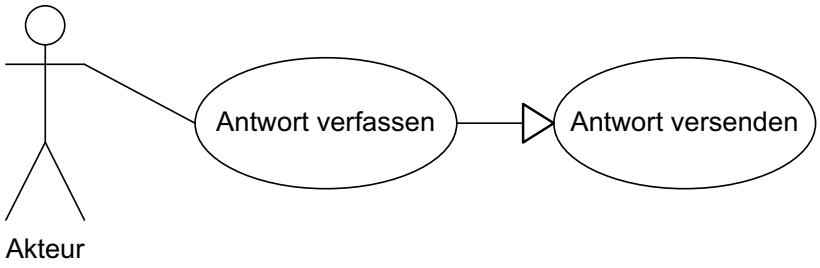


Abbildung 3.5: Anwendungsfalldiagramm Beantworten von Mitteilungen

Vor- und Nachbedingung: Der Akteur ist im System registriert und angemeldet. Am Ende des Anwendungsfalls ist die Mitteilung dem Empfänger zugestellt worden und der Benutzer erhält eine aktualisierte Liste seiner versendeten Antworten.

Beschreibung: Der Akteur benutzt die Internetanwendung und meldet sich durch Eingabe seines Nutzernamens und seines Passworts beim System an. Er hat Zugriff auf aktuellen Mitteilungen, die ihm aufgrund seiner Profileinstellungen zugeordnet worden sind und auf alle die im System verfügbar sind. Er wählt eine Mitteilung aus und bekommt die Details der Mitteilung angezeigt. Der Akteur antwortet dem Verfasser der Mitteilung, indem er eine Antwort erstellt. Hierfür werden keine zusätzlichen Angaben wie Titel oder Gültigkeitsdauer benötigt, sondern lediglich der Antworttext. Anschließend wird die Mitteilung versendet. Die Antwort wird nun dem Empfänger zugestellt und der Akteur erhält eine aktualisierte Liste seiner versendeten Antworten. Im Gegensatz zum Veröffentlichen von Mitteilungen in Abschnitt 3.4.1.1 wird eine Antwort nicht indiziert oder kategorisiert, da sie an einen bestimmten Empfänger und nicht an Eigenschaften eines beliebigen Empfängers gerichtet ist.

3.4.1.4 Profil bearbeiten

Dieser Anwendungsfall beschreibt das Bearbeiten des Benutzerprofils.

Vor- und Nachbedingung: Der Akteur ist im System registriert und angemeldet. Am Ende des Anwendungsfalls ist das aktualisierte Benutzerprofil im System gespeichert.

Beschreibung: In diesem Anwendungsfall werden zwei Arten von Profilen unterschieden, *Statisches Profil* und *Dynamisches Profil*. Beide Profilarten beziehen sich auf die Eigenschaften des Benutzers. Der Akteur benutzt die Internetanwendung und meldet sich durch Eingabe seines Nutzernamens und seines Passworts am System an.

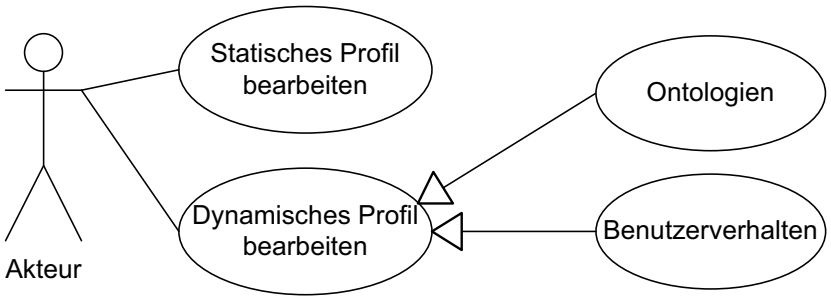


Abbildung 3.6: Anwendungsfalldiagramm Profil bearbeiten

Statisches Profil: Auf Grundlage der bei der Registrierung angegebenen Daten wurde ein statisches Profil für den Benutzer angelegt, das allgemeine Angaben zur Person und deren Tätigkeit beinhaltet. Dieses Profil kann nur durch Eingaben des Benutzers verändert werden.

Dynamisches Profil: Das dynamische Profil besteht aus den Interessen des Benutzers und ist nach der Registrierung zunächst leer. Der Akteur wählt ein Interesse aus bestehenden Ontologien aus. Dieses Interesse wird in sein dynamisches Profil übernommen. Anschließend legt er den Grad seines Interesses fest, also wie sehr er sich für die gewählte Kategorie interessiert. Der Grad des Interesses kann explizit vom Benutzer selbst und implizit durch sein Verhalten (Bewertung von Mitteilungen) verändert werden.

Der Akteur hat die Daten in seinem Profil verändert und erhält ein aktualisiertes Benutzerprofil. Die im System vorhandenen, aktuellen Mitteilungen werden nun aufgrund des neuen Profils auf Übereinstimmung überprüft und der Benutzer erhält eine aktualisierte Liste von Vorschlägen (siehe Abschnitt 3.4.1.2).

3.4.2 Systemabläufe

In diesem Abschnitt werden die aus den funktionalen Anforderungen in Abschnitt 3.4.1 abgeleiteten Systemabläufe vorgestellt.

Abbildung 3.7 zeigt den allgemeinen Ablauf des Systems. Nach anzeigen der Startseite und erfolgreicher Benutzeranmeldung bekommt der Benutzer eine personalisierte Seite angezeigt. Auf dieser Seite hat er die Möglichkeit, auf die verschiedenen Funktionen des Systems zuzugreifen. Er kann eine Mitteilung anzeigen, eine Mitteilung erstellen und sein Profil bearbeiten.

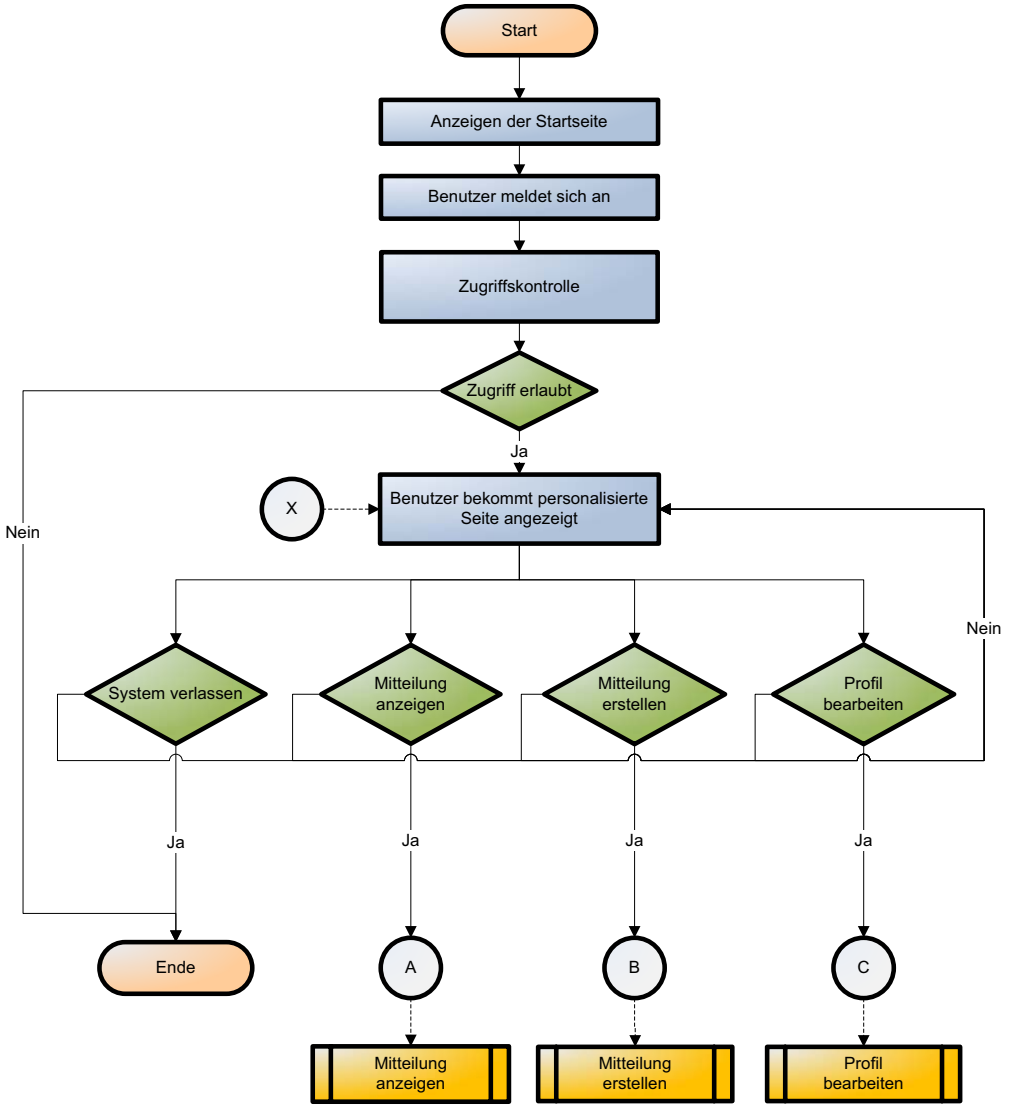


Abbildung 3.7: Flussdiagramm — Allgemeiner Programmablauf

3.4.2.1 Anzeigen von Mitteilungen

Abbildung 3.8 zeigt das Anzeigen einer Mitteilung. Der Benutzer bekommt eine Maske angezeigt, in der er eine Mitteilung anzeigen kann. Dies kann sowohl eine zugeordnete oder eine aktuelle Mitteilung oder eine erhaltene Antwort sein. Er hat die Möglichkeit auf diese Mitteilung zu antworten und wird anschließend auf den Ablauf „Mitteilung erstellen“ verwiesen. (Siehe Anwendungsfall 3.4.1.3.)

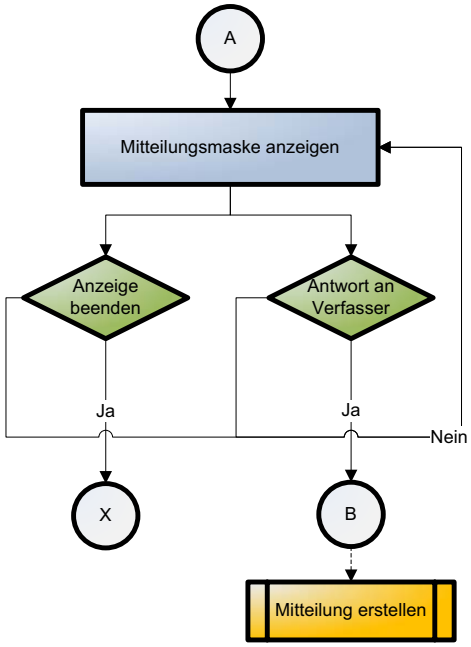


Abbildung 3.8: Flussdiagramm - Anzeigen von Mitteilungen

3.4.2.2 Mitteilung erstellen

Abbildung 3.9 zeigt das Erstellen einer Mitteilung. (Siehe Anwendungsfall 3.4.1.1.) An dieser Stelle wird unterschieden, ob es sich um eine neue Mitteilung oder eine Antwort auf eine Mitteilung handelt.

Wenn es sich um eine Antwort auf eine bereits erhaltene Mitteilung handelt, kann der Benutzer den Mitteilungsbetreff bearbeiten und den Inhalt der Mitteilung verfassen. Anschließend wird die Mitteilung veröffenticht und dem Verfasser der ursprünglichen Mitteilung zugesandt.

Bei einer neuen Mitteilung kann der Benutzer die Gültigkeit der Mitteilung angeben, den Mitteilungsbetreff eingeben und den Inhalt der Mitteilung verfassen. Anschließend kann die Mitteilung kategorisiert und vom Benutzer bearbeitet werden. Er kann sowohl Kategorien aus- als auch neue Kategorien einschließen. Anschließend wird die Mitteilung veröffentlicht.

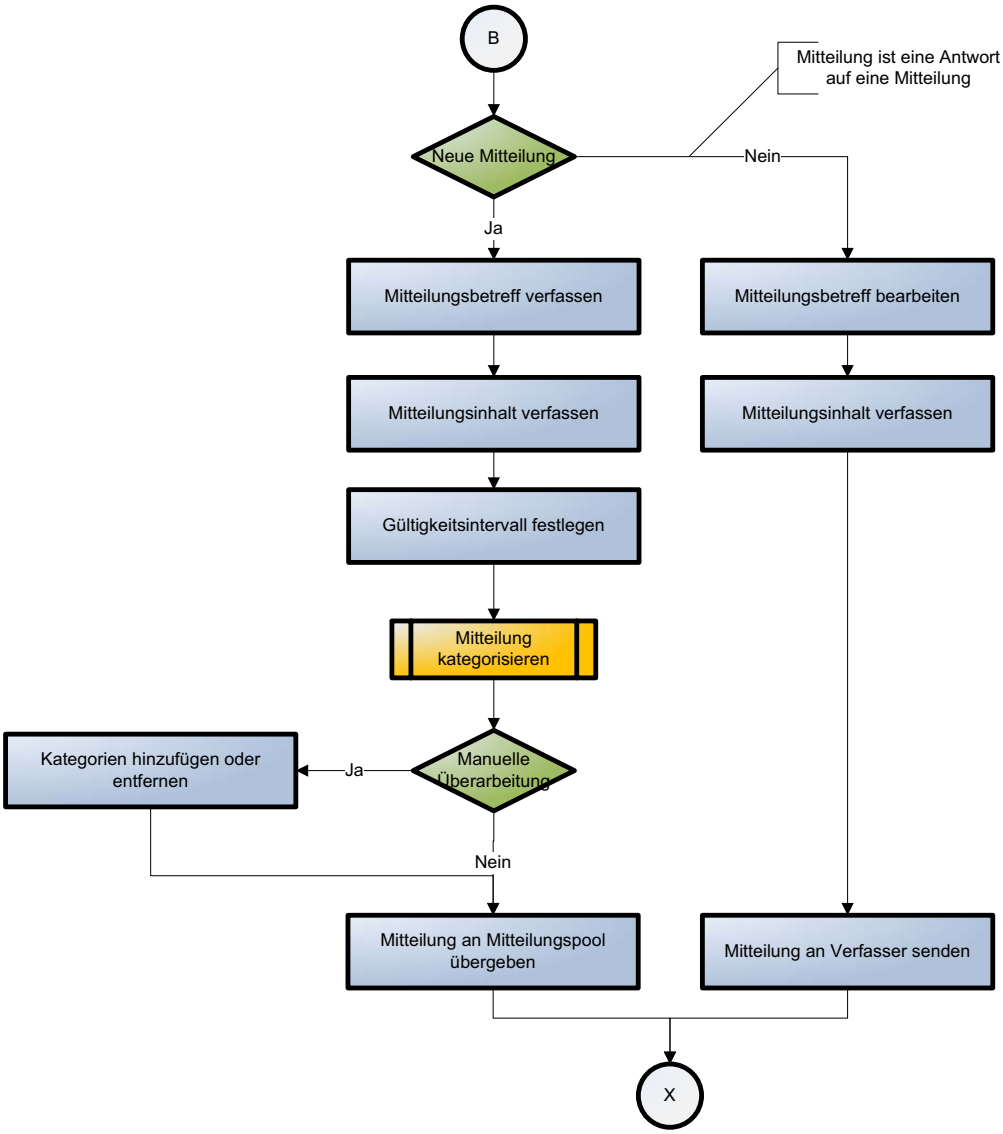


Abbildung 3.9: Flussdiagramm — Erstellen einer Mitteilung

3.4.2.3 Profil bearbeiten

Abbildung 3.10 zeigt das Bearbeiten des persönlichen Profils. (Siehe Anwendungsfall 3.4.1.4.) Sollte noch kein Profil zur Verfügung stehen, wird ein neues Profil erstellt. Anschließend hat der Benutzer die Möglichkeit einen neuen Eintrag hinzu zu fügen oder einen bestehenden Eintrag zu ändern oder zu entfernen. Im Anschluß daran wird das Profil des Benutzers aktualisiert.

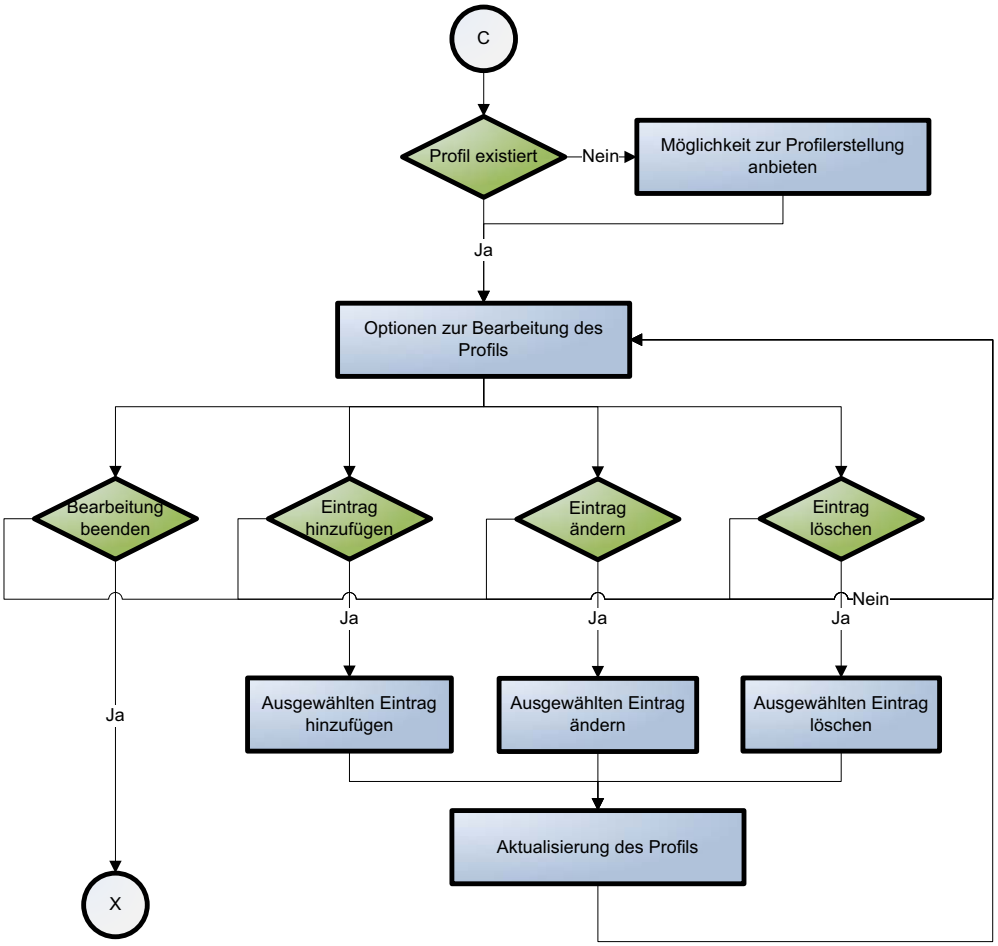


Abbildung 3.10: Flussdiagramm — Bearbeiten des persönlichen Profils

3.5 Nichtfunktionale Anforderungen

Neben den in Abschnitt 3.4 beschriebenen funktionalen Anforderungen an ein System gibt es auch nichtfunktionale Anforderungen. Diese richten sich an die Umstände, unter denen die geforderten Funktionen zu erfüllen sind. Nichtfunktionale Anforderungen können ebenfalls kritisch für den Erfolg einer Anwendung sein, da sie die Qualitätseigenschaften betreffen. Die ISO/IEC 9126 definiert sechs Qualitätsmerkmale für Software-Produkte. Dazu gehören Anforderungen an die Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Wartbarkeit und Übertragbarkeit.

3.5.1 Funktionalität

Hierunter versteht man die Eignung des Systems, die geforderten Funktionen zu erfüllen.

Die Anwendung soll dem Benutzer die Möglichkeit bieten, ein personalisiertes Profil zu erstellen. Der Benutzer soll das Profil innerhalb der Vorgaben bearbeiten können. Auf Benutzereingaben soll ein korrektes Ergebnis geliefert werden und die Anwendung soll angemessene Funktionen für die Aufgabenerfüllung bereitstellen. Die Authentifizierung des Benutzers und die Integrität der Daten soll gegeben sein.

3.5.2 Zuverlässigkeit

Die Fähigkeit des Systems, ein bestimmtes Leistungsniveau über einen bestimmten Zeitraum aufrecht zu halten.

Die Qualität der Daten muss gewährleistet sein und der Dienst soll für den Benutzer zuverlässig zur Verfügung stehen. Er sollte 24 Stunden am Tag, 7 Tage die Woche und 365 Tage im Jahr erreichbar sein und eine Ausfallzeit von 100 Stunden pro Jahr nicht übersteigen.

3.5.3 Benutzbarkeit

Hier werden Kriterien definiert, die für die Benutzung und die individuelle Beurteilung durch eine bestimmte Benutzergruppe erforderlich sind.

Der Benutzer soll die Möglichkeit haben, das Systemkonzept und die Anwendung zu verstehen. Er soll die Bedienung und Steuerung der Anwendung durch möglichst einfache Strukturen intuitiv erlernen können. Die Attraktivität der Anwendung soll für den Benutzer durch den Einsatz aktueller Web-Elemente gesteigert werden.

3.5.4 Effizienz

Effizienz meint die Eignung des Systems, ein angemessenes Leistungsniveau im Verhältnis zu den eingesetzten Betriebsmitteln bereit zu stellen.

Bei der Ausführung der Anwendung sollen die Antwort- und Verarbeitungszeiten in angemessener Zeit erfolgen. Die Anfrage des Client-Anwendung an die Server-Anwendung soll asynchron erfolgen, so dass die Client-Anwendung vom Benutzer trotz Anfrage weiter bedient werden kann und nicht blockiert. Die Antwort der Server-Anwendung kann je nach Anfrage und Übertragungsgeschwindigkeit variieren. Die durchschnittliche Antwortzeit beträgt 2 Sekunden, darf aber 60 Sekunden nicht überschreiten.

3.5.5 Wartbarkeit

Wartbarkeit fordert die Möglichkeit, dass das System in Bezug auf Korrekturen, Verbesserungen und Anpassungen modifiziert werden kann.

Auftretende Mängel oder Ursachen für ein Versagen der Anwendung sollen erkennbar sein. Anpassungen oder Fehlerbehandlungen sollen kostengünstig möglich sein und die Testbarkeit des Systems muss gewährleistet sein. Hierzu sollen die Verantwortlichkeiten der Systemkomponenten erkennbar und dokumentiert sein. Für die einzelnen Komponenten müssen Regressionstests zur Verfügung stehen.

3.5.6 Übertragbarkeit

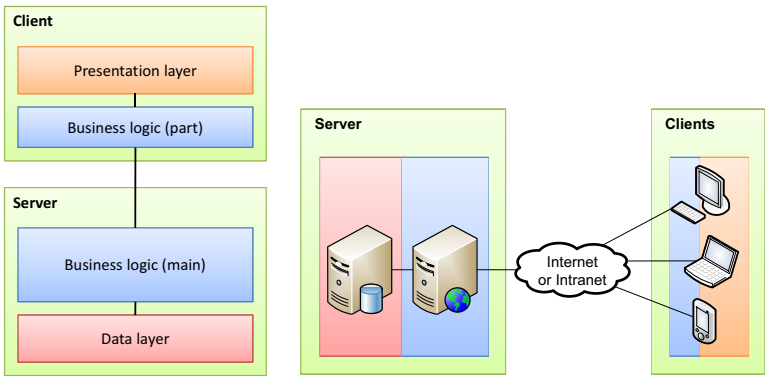
Das System sollte von einer Umgebung in eine andere übertragen werden können. Die Umgebung kann eine organisatorische, Hardware- oder Software-Umgebung sein.

Bei der Anwendung handelt es sich um eine Webanwendung, die auf einem Apache Tomcat Server laufen soll. Hierdurch soll eine Plattformunabhängigkeit gewährleistet werden.

4 Systemarchitektur

In diesem Kapitel wird die Architektur des erstellen Prototyps (Profil.Info.Sys) beschrieben. Der Prototyp ist eine Webanwendung und basiert auf einer Client-Server-Architekur. Für die Kommunikation der beiden Systemteile werden GWT-Remote Procedure Calls eingesetzt.[1] In Abschnitt 4.1 wird zunächst die Server-Architektur beschrieben. Anschließend wird in Abschnitt 4.2 die Client-Architektur erläutert.

Das System basiert auf einer konzeptionellen Drei-Schichten-Architektur. Die einzelnen Schichten (Präsentations-, Anwendungs- und Datenhaltungsschicht) spiegeln hierbei die grundsätzlichen Aufgaben des Software-Systems wider.
Abbildung 4.1(a) zeigt die Verteilung der einzelnen Schichten und Abbildung 4.1(b) einen beispielhaften Aufbau der Anwendung.



(a) Verteilungsmuster (b) Beispiel

Abbildung 4.1: Verteilung der Anwendung

Es ist zu sehen, dass der Client sowohl die Präsentationsschicht als auch Teile der Anwendungsschicht bzw. Anwendungslogik enthält. Der Webbrowser der einzelnen Geräte ist der Client der Anwendung. Die Anwendung ist für mehrere Benutzer gleichzeitig nutzbar und netzwerkfähig. Der Server beinhaltet den Hauptteil der Anwendungslogik und ist für die Datenhaltung verantwortlich.

[1] Siehe Grundlagenkapitel 2.3.1.

4.1 Server-Architektur

Der Server ist für die Datenhaltung und den Hauptteil der Anwendungslogik verantwortlich. Er nimmt Daten über eine definierte Schnittstelle (Dienstschicht) von den Clients entgegen und stellt ihnen hierüber neue Daten zur Verfügung. Die konzeptionelle Sicht auf die Architektur ist in Abbildung 4.2 dargestellt. Abschnitt 4.1.1 erläutert die einzelnen Komponenten dieser Architektur. Anschließend wird in Abschnitt 4.1.2 das Zusammenspiel der Server-Komponenten erläutert.

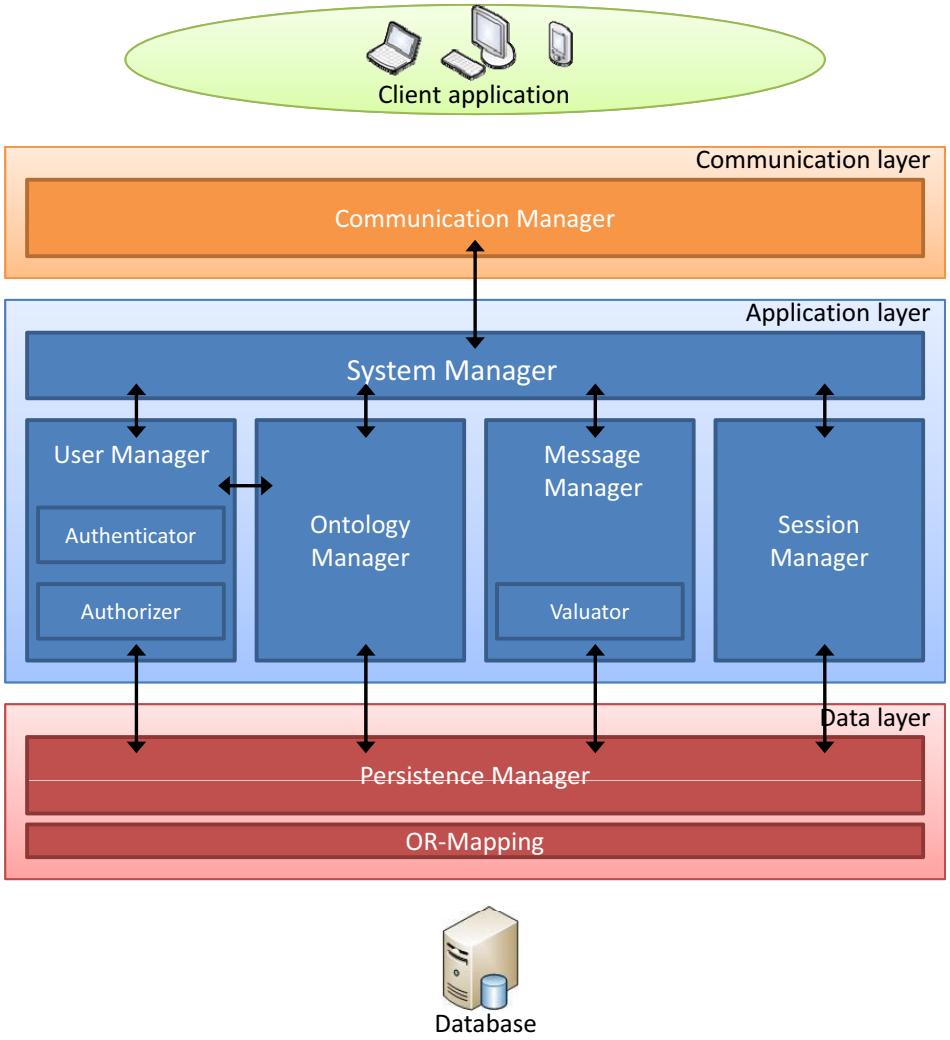


Abbildung 4.2: Konzeptionelle Sicht auf die Server-Architektur

4.1.1 Komponenten

Die Architektur des Servers besteht im Wesentlichen aus drei Schichten. Die oberste Schicht ist die Kommunikationsschicht (**Communication layer**), die den *Communication Manager* bein-

haltet. Sie bildet den Einstiegspunkt für alle Konsumenten und fasst die Dienste der Anwendung zusammen.

Die mittlere Schicht ist die Anwendungsschicht (**Application layer**) und enthält die serverseitige Anwendungslogik. Sie besteht aus dem *System Manager*, dem *Ontology Manager*, dem *User Manager*, dem *Message Manager* und dem *Session Manager*.
Die einzelnen Komponenten dieser Schicht kümmern sich jeweils um einen bestimmten fachlichen Bereich der Anwendungslogik. Der *SystemManager* orchestriert die einzelnen Komponenten. Er delegiert Anfragen, die den Benutzer des Systems direkt betreffen, an den *UserManager*. Anfragen an Gesamt- und Unter-Ontologien werden an den *OntologyManager* weitergeleitet. Die Aufgaben für Mitteilungen und deren Bewertung werden an den *MessageManager* abgegeben und der *SessionManager* wird bei der Erzeugung und Validierung der Session gerufen. Durch diese Aufteilung werden die Verantwortlichkeiten des Systems voneinander getrennt.

Die unterste Schicht ist die Datenschicht (**Data layer**). Sie ist für die Kommunikation mit der Datenbank zuständig. Zu ihr gehören die Komponenten *Persistence Manager* und *OR-Mapping*.

Fachliche Komponenten

- **Communication Manager**
  Der Communication Manager ist für die Kommunikation mit den Clients zuständig und fasst die angebotenen Dienste zusammen.
- **System Manager**
  Der System Manager ist der zentrale Zugriffspunkt der Server-Anwendung. Er ist für die Aufbereitung der Ergebnisse zuständig und kommuniziert mit den darunter liegenden Komponenten.
- **User Manager**
  Der User Manager nimmt alle Anfragen, die den Benutzer direkt betreffen, entgegen. Er kümmert sich sowohl um das Benutzerprofil als auch um die Benutzerverwaltung. Bei Anfragen an das dynamische Profil kommuniziert er mit der Ontology Manager-Komponente. Er beinhaltet die *Authenticator*-Komponente, die für die Benutzerüberprüfung zuständig ist und die *Authorizer*-Komponente, die für die Überprüfung der Benutzerberechtigung vorgesehen ist.
- **Ontology Manager**
  Der Ontology Manager ist für die Verwaltung der Ontologien zuständig. Er hat Zugriff auf die Gesamt-Ontologien und kümmert sich um die Aufbereitung der jeweiligen Unter-Ontologien der Benutzer. Die Unter-Ontologien repräsentieren das dynamische Profil und damit die Interessen eines Benutzers. Die Funktionalität zum Erstellen und Ändern dieses Profils wird in dieser Komponente gekapselt.
- **Message Manager**
  Der Message Manager ist für die Verwaltung der Mitteilungen zuständig und beinhaltet

die *Valuator*-Komponente, die für das Indizieren und Bewerten von Mitteilungen zuständig ist.

- **Session Manager**
  Der Session Manager ist für die Erzeugung und Verwaltung der Benutzer-Session zuständig.

Technische Komponenten

- **Persistence Manager**
  Der Persistence Manager ist für die Kommunikation mit der Datenbank bzw. mit der *Hibernate* Komponente zuständig. Er dient als Zugriffspunkt für die Komponenten der darüber liegenden Anwendungsschicht.
- **OR-Mapping**
  Die OR-Mapping Komponente ist für die Anbindung der relationalen Datenbank an das objektorientierte System zuständig und bietet eine Abstraktion von der konkret verwendeten Datenbank.[2]

4.1.2 Zusammenspiel der Komponenten

Das Zusammenspiel der einzelnen Komponenten wird in der Abbildung 4.2 durch Pfeile visualisiert.

- **Communication Manager**
  Der Communication Manager nimmt alle Aufrufe der Service-Implementationen entgegen und leitet diese direkt an den System Manager weiter. Die entsprechenden Ergebnisse werden anschließend an den Service zurückgegeben, der sich um die Nachrichtenverarbeitung und das Serialiseren der zu übertragenden Objekte kümmert.
- **System Manager**
  Der System Manager greift direkt auf die Methoden des Ontology Managers, des User Managers, des Message Managers und des Session Managers zu. Er bereitet die Ergebnisse auf und gibt ein entsprechendes Ergebnis an den Communication Manager zurück.
- **User Manager**
  Der User Manager greift bei Anfragen an das dynamische Benutzerprofil auf den Ontology Manager zu. Anfragen an das statische Profil werden selbst behandelt. Er beinhaltet den Authenticator und den Authorizer und kommuniziert direkt mit diesen Klassen. Bei Bedarf greift er auf den Persistence Manager zu. Die Ergebnisse werden an den System Manager zurück gegeben.

[2] In dieser Arbeit wurde hierfür Hibernate Version 3.2.4 SP1 verwendet. Siehe Kapitel 5.2.

- **Ontology Manager**
  Der Ontology Manager kommuniziert ebenfalls direkt mit dem Persistence Manager. Die von ihm berechneten Ergebnisse werden wieder an den System Manager zurück gegeben.
- **Message Manager**
  Der Message Manager beinhaltet den Valuator. Er kommuniziert direkt mit dieser Klasse und greift bei Bedarf auf den Persistence Manager zu. Die Ergebnisse werden anschließend an den System Manager zurückgegeben.
- **Session Manager**
  Der Session Manager kommuniziert ebenfalls direkt mit dem Persistence Manager. Er gibt die von ihm berechneten Ergebnisse an den System Manager zurück.
- **Persistence Manager**
  Der Persistence Manager greift über die OR-Mapping-Komponente auf die Datenbank zu und legt bei Bedarf neue Datensätze an, ändert oder entfernt diese. Anschließend gibt er die entsprechenden Ergebnisse an den System Manager zurück.

4.2 Client-Architektur

Der Client visualisiert die Anwendung und kommuniziert mit dem Server. Abbildung 4.3 zeigt die konzeptionelle Sicht auf die Client-Architektur. In Abschnitt 4.2.1 werden die einzelnen Komponenten dieser Architektur erläutert. Abschnitt 4.2.2 erläutert das Zusammenspiel der Client-Komponenten.

4.2.1 Komponenten

Die Architektur des Clients besteht aus drei Schichten. Die oberste Schicht ist die Präsentationsschicht (**Presentation layer**). Sie ist für die Visualisierung der Anwendung zuständig. Sie ist eine Webanwendung und wird durch die Verwendung der *GUI* Komponente realisiert. Ein Fokus dieser Arbeit liegt auf der Benutzbarkeit und intuitiven Bedienbarkeit des Systems. Daher wird in dieser Komponente besonderer Wert auf die Beachtung der Ergonomiegrundsätze nach DIN EN ISO 9241-110:2006 [ISO06] gelegt. Auf die einzelnen Maßnahmen wird in Kapitel 5 ausführlicher eingegangen.

Die mittlere Schicht ist die Anwendungsschicht (**Application layer**), die sich um die clientseitige Logik der Anwendung kümmert. Sie besteht aus dem *Portal Manager*, dem *User Manager*, dem *Message Manager*, dem *Ontology Manager* und dem *Session Manager*.
Aufgrund der asynchronen Kommunikation mit dem Server besitzen die Komponenten einen Teil der Anwendungslogik. So kann z.B. bei der Abmeldung eines Benutzers die Session ungültig gemacht und der Benutzer abgemeldet werden, auch wenn noch keine Rückmeldung des

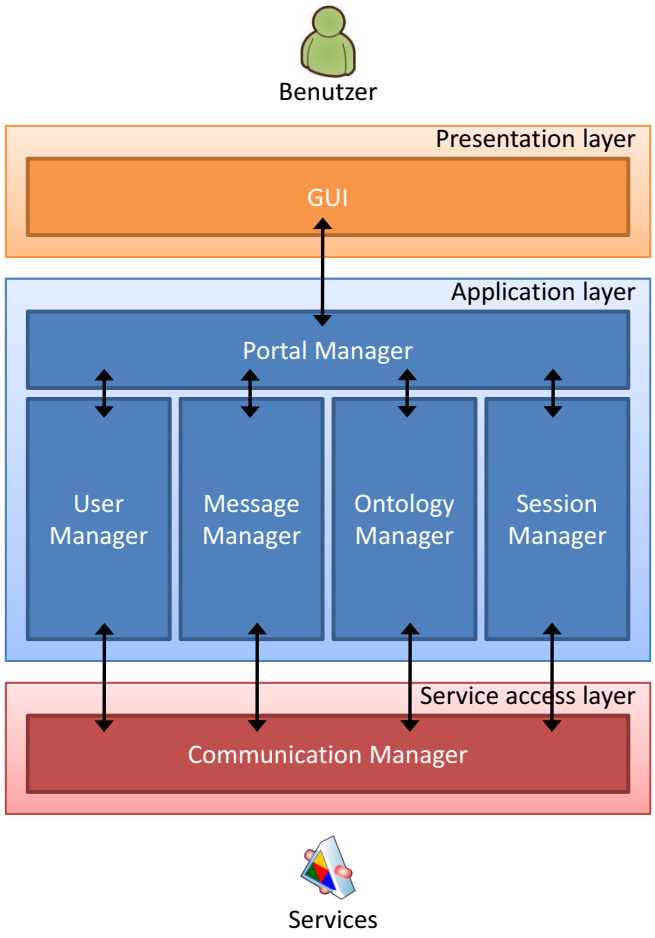


Abbildung 4.3: Konzeptionelle Sicht auf die Client-Architektur

Servers erfolgt ist. Die Verantwortlichkeiten des Systems werden durch die einzelnen Komponenten voneinander getrennt. Der *PortalManager* delegiert die Aufgaben an die entsprechenden Komponenten. Der *UserManager* kümmert sich um Anfragen, die den Benutzer des Systems betreffen. Der *MessageManager* wird für Aufgaben, die Mitteilungen betreffen, verwendet. Bei Anfragen an Ontologien wird der *OntologyManager* verwendet und bei Aufgaben wie z.B. beim Anfordern oder Überprüfen einer Session wird der *SessionManager* eingesetzt. Bei auftretenden Veränderungen in einem der Verantwortungsbereiche benachrichtigen die betroffenen Komponenten den *PortalManager*.

Die untere Schicht ist die Dienstzugriffsschicht (**Service access layer**). Sie ist für die Kommunikation mit dem Server zuständig. Sie beinhaltet die *Communication Manager* Komponente und bietet der Anwendungsschicht einen Zugriff auf die entsprechenden Dienste des Servers.

Fachliche Komponenten

- **GUI**
  Die GUI-Komponente ist die grafische Benutzeroberfläche, die die Webanwendung visualisiert. Sie ermöglicht dem Benutzer eine Interaktion mit der Anwendung. Da es sich

in dieser Arbeit um eine Webanwendung handelt, erfolgt die Bedienung des Systems über Webseiten mittels eines Webbrowsers.[3] Durch eine Auswahl von geeigneten Darstellungsformen soll der Benutzer bei der Erfüllung seiner Aufgaben unterstützt werden. Ihm soll ein komfortabler Zugang zu Mitteilungen, eine verständliche Navigationsstrukur und eine intuitive Nutzung des Systems ermöglicht werden. (Siehe Abschnitt 1.2)

- **Portal Manager**
  Der Portal Manager ist der zentrale Zugriffspunkt der Client Anwendung. Er kapselt die Funktionalität der Anwendung und kommuniziert mit den darunter liegenden Komponenten.
- **User Manager**
  Der User Manager ist für die Benutzerverwaltung zuständig. Er verwaltet den angemeldeten Benutzer und dessen Eigenschaften und Mitteilungen.
- **Message Manager**
  Der Message Manager ist für die Verwaltung der Mitteilungen zuständig, die nicht direkt den angemeldeten Benutzer betreffen. Er kümmert sich um die aktuell im System verfügbaren Mitteilungen.
- **Ontology Manager**
  Der Ontology Manager ist für die Verwaltung der Ontologien zuständig. Er kümmert sich um das Laden der verfügbaren „Gesamt“ Ontologien.
- **Session Manager**
  Der Session Manager ist für die Verwaltung der Benutzer-Session zuständig.

Technische Komponenten

- **Communication Manager**
  Der Communication Manager ist für die Kommunikation mit dem Server zuständig. Die entfernten Aufrufe erfolgen mittels GWT-RPC. (Siehe Kapitel 2.3.1.2)

4.2.2 Zusammenspiel der Komponenten

Das Zusammenspiel der einzelnen Komponenten wird in der Abbildung 4.3 durch Pfeile visualisiert.

- **GUI**
  Die GUI-Komponente besteht aus mehreren Java-Klassen, die direkt auf den Portal Manager der Anwendungsschicht zugreifen. Zur Aktualisierung der Daten implementieren die entsprechenden Klassen das Observer-Interface und werden so durch den Portal Manager über Ergebnisse und Änderungen informiert.

[3] In dieser Arbeit wurde das Google Web Toolkit Version 1.4.59 verwendet. Siehe Kapitel 5.2. Die Widget-Komponenten des GWT werden für die Visualisierung der Webanwendung verwendet. Der eigentliche Java-Code wird in Browserkonformes JavaScript und HTML übersetzt. (Siehe Kapitel 2.3.1)

- **Portal Manager**
  Der Portal Manager ist der zentrale Zugriffspunkt für die Präsentationsschicht. Er greift auf die Funktionen des User Managers, des Message Managers, des Ontology Managers und des Session Managers zu, die er nicht selbst bereitstellt und macht diese Funktionen nach außen verfügbar. Er erweitert die Observable Klasse, wodurch die Zuhörer über Ergebnisse informiert werden können. Zudem implementiert er das Observer Interface, um über Ergebnisse des User Managers, des Message Managers, des Ontology Managers und des Session Managers informiert zu werden.

- **User Manager**
  Der User Manager erweitert die Observable-Klasse um den Portal Manager über Ereignisse, die den angemeldeten Benutzer betreffen, zu informieren. Er informiert den Portal Manager z.B. über Benutzerregistrierungen, Änderungen an Benutzereigenschaften und Mitteilungen, die zu den Interessen des Benutzers passen. Er greift auf die entsprechenden Methoden des Communication Managers zu und implementiert das Observer Interface um die Ergebnisse des Communication Managers zu erhalten.

- **Message Manager**
  Der Message Manager erweitert die Observable-Klasse um den Portal Manager über Ereignisse zu informieren. Er informiert diesen z.B. über die im System verfügbaren Mitteilungen und über vorläufige Kategorisierungen von Mitteilungen. Er greift auf die entsprechenden Methoden des Communication Managers zu und implementiert das Observer Interface, um dessen Ergebnisse zu erhalten.

- **Ontology Manager**
  Der Ontology Manager erweitert die Observable-Klasse um den Portal Manager über die aktuell verfügbaren „Gesamt“-Ontologien zu informieren. Er greift über den Communication Manager auf die entsprechende Methode zu und implementiert das Observer Interface um das Ergebnis des Communication Managers zu erhalten.

- **Session Manager**
  Der Session Manager erweitert die Observable-Klasse um den Portal Manager über Ereignisse, die die Benutzer-Session betreffen, zu informieren. Nach einem erfolgreichen Login wird dieser z.B. über eine gültige Session informiert. Er greift auf die entsprechenden Methoden des Communication Manager zu und implementiert das Observer Interface um dessen Ergebnisse zu erhalten.

- **Communication Manager**
  Der Communication Manager erweiter die Observable Klasse, um seine Zuhörer über die gewünschten Ergebnisse zu informieren. Er ruft die asynchronen Methoden der entsprechenden Dienste auf und informiert über die Ergebnisse bei Erfolg oder Misserfolg der durchgeführten Aufrufe.

5 Systemimplementierung

In diesem Kapitel wird die Anwendung aus Implementierungssicht beschrieben. In Abschnitt 5.1 wird die Implementierung für den Server- und Client-Teil sowie das Anwendungsmodell diskutiert. Abschnitt 5.2 stellt die zur Entwicklung verwendeten Software-Komponenten vor. Abschließend beschreibt Abschnitt 5.3 die zur Qualitätssicherung eingesetzten Maßnahmen.

5.1 Anwendung

In diesem Abschnitt werden die Komponenten der herausgearbeiteten Architekturen detailliert beschrieben. Abschnitt 5.1.1 beschreibt die Server- und Abschnitt 5.1.2 die Client-Anwendung aus der Implementierungssicht.

5.1.1 Server-Anwendung

Dieser Abschnitt erläutert die Komponenten der in Abschnitt 4.1 vorgestellten Server-Architektur. Die einzelnen Schichten werden durch die Abschnitte 5.1.1.1 für die Dienstschicht (Service layer), Abschnitt 5.1.1.2 für die Anwendungsschicht (Application layer) und Abschnitt 5.1.1.3 für die Datenschicht (Data layer) beschrieben. Anschließend wird die zugrunde liegende Datenbank in Abschnitt 5.1.1.4 erläutert.

Abbildung 5.1 zeigt das Klassendiagramm (Komponenten) der Server Anwendung.

5.1.1.1 Communication layer

Die Kommunikationsschicht (Communication layer) ermöglicht den Clients den Zugriff auf die Dienste des Servers. Die Klassen `MessageServiceImpl`, `OntologyServiceImpl` und `UserServiceImpl` implementieren die jeweiligen Schnittstellen der Client Interfaces `MessageService`, `OntologyService` und `UserService`. Sie erweitern das `RemoteServiceServlet`, das sich um die Nachrichtenverarbeitung und das Serialisieren der übertragenen Objekte kümmert.[1]

[1] Vergleiche Grundlagenkapitel 2.3.1.

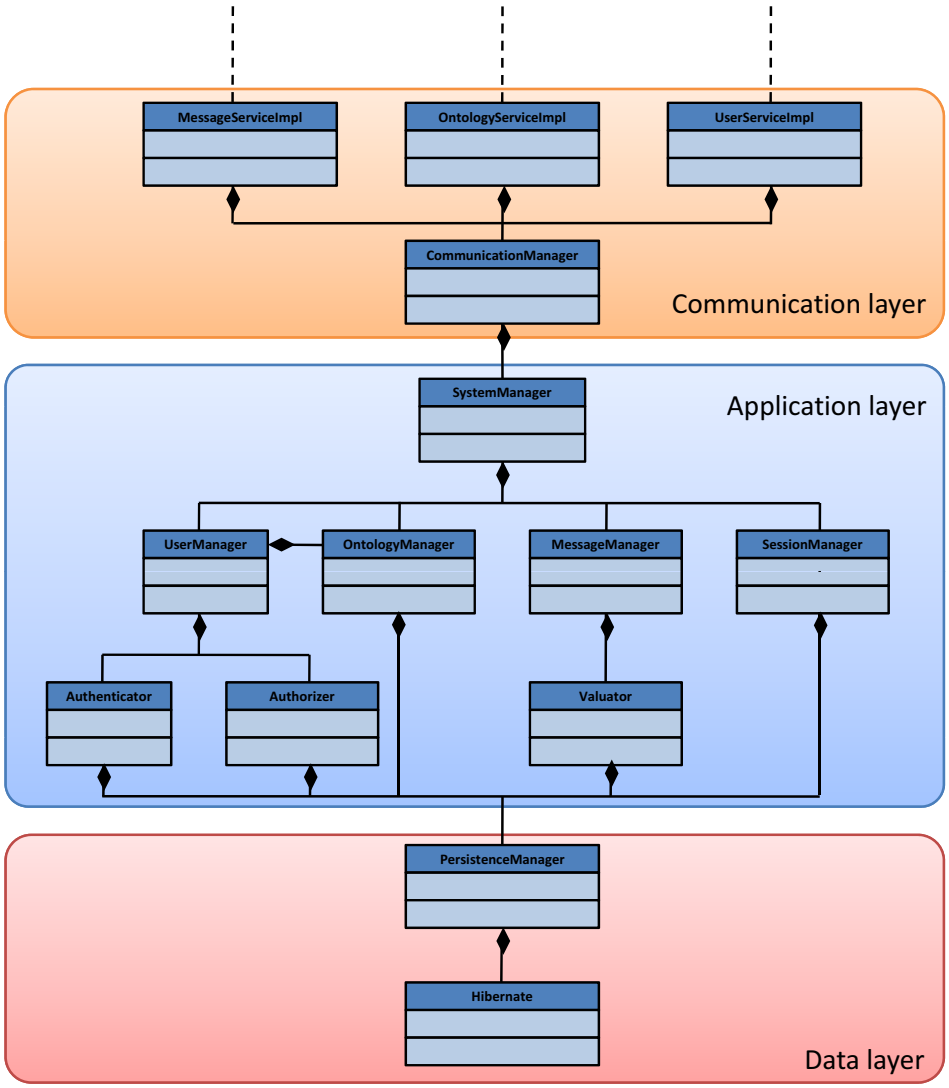


Abbildung 5.1: Klassendiagramm — Komponenten der Server Anwendung

Diese Service-Implementationen richten sich direkt an den `CommunicationManager`, der alle Aufrufe entgegen nimmt. Die Aufrufe werden anschließend an die Anwendungsschicht bzw. an die Klasse `Systemmanager` weitergeleitet.

5.1.1.2 Application layer

Die Anwendungsschicht (Application layer) beinhaltet die serverseitige Anwendungslogik der Anwendung. Die Service-Aufrufe werden von der Dienstschicht bzw. von der Klasse `CommunicationManager` an die `Systemmanager`-Klasse weitergeleitet. Diese ist zentraler Zugriffspunkt für die Kommunikationsschicht und ruft die für die Abarbeitung benötigten Methoden der entsprechenden Klassen auf. Die erhaltenen Ergebnisse werden bei Bedarf aufbereitet und überprüft.

UserManager-Komponente

Die Klasse `UserManager` beinhaltet die Klassen `Authenticator` und `Authorizer`. Sie nimmt alle Anfragen, die den Benutzer betreffen, entgegen und stellt entsprechende Methoden nach außen zur Verfügung. Anfragen an das dynamische Benutzerprofil werden an den `OntologyManager` delegiert und Anfragen, die das statische Profil betreffen, werden selbst behandelt.

Die Authentifikation eines Benutzers wird von der Klasse `Authenticator` übernommen. Hier wird z.B. überprüft, ob sich ein Benutzer korrekt am System angemeldet hat und ob ein gewählter Benutzername noch zu vergeben ist.

Die Autorisierungsprüfung übernimmt die Klasse `Authorizer`. Sie überprüft, welche Rechte der angemeldete Benutzer im System hat.

Operationen wie Laden und Ändern von Benutzern werden vom `UserManager` an den `PersistenceManager` delegiert.

OntologyManager-Komponente

Die Klasse `OntologyManager` behandelt alle Anfragen, die Ontologien betreffen. Bei diesen kann es sich sowohl um Anfragen für Gesamt-Ontologien als auch um Anfragen zu Unter-Ontologien (Interessen eines Benutzers) handeln.

Die im System verfügbaren Gesamt-Ontologien sind vorgegeben und werden nicht von dieser Klasse erstellt. Zur Erstellung der Ontologien wurde ein Ontologie-Editor verwendet.[2] Die Ontologien sind OWL-DL-Ontologien (Siehe Abschnitt 2.2.4) und liegen im `<ontologieName>.owl` Format vor. Mit Hilfe der OWL-Bibliothek „OWL API“[3] werden die Ontologien eingelesen.

Bei einer Anfrage nach verfügbaren Gesamt-Ontologien werden `OWLOntology`-Klassen in `Ontology`-Klassen konvertiert, da die `OWLOntologien` nicht serialisiert und damit nicht für die RPC-Übertragung verwendet werden können. Die Klasse `Ontology` besteht aus dem Namen der zugehörigen Ontologie, dem Namen der Klasse und den zugehörigen Ober- bzw. Unterklassen. Somit bleibt die Struktur erhalten und die Ontologien können im Client visualisiert werden. (Siehe Abschnitt 5.1.2.1.)

Sobald ein Benutzer sein Profil aktualisiert und ein Interesse auswählt, wird eine Unter-Ontologie erstellt. Abbildung 5.2 visualisiert dies an einer einfachen Beispiel-Ontologie[4]. Die Unter-Ontologie ist ein minimaler Teilbaum der Gesamt-Ontologie. Den Instanzen der Klassen wird eine Eigenschaft (DataProperty) hinzugefügt, die den Wert des Interesses an der Klasse repräsentiert. Das ausgewählte Interesse hat in diesem Beispiel einen Interessenswert von 80 Prozent.

[2] In dieser Arbeit wurde Protégé 3.3.1 (build 430) verwendet. Siehe Kaptiel 5.2.
[3] In dieser Arbeit wurde OWL API 2.1.1 verwendet. Siehe Kapitel 5.2.
[4] Die Beispiel-Ontologie wurde auch für die Funktionstests in Kapitel 5.3 verwendet.

Für die beiden Oberklassen wird ein Wert von 0 Prozent angenommen, da hierüber noch keine Angaben des Benutzers vorhanden sind.

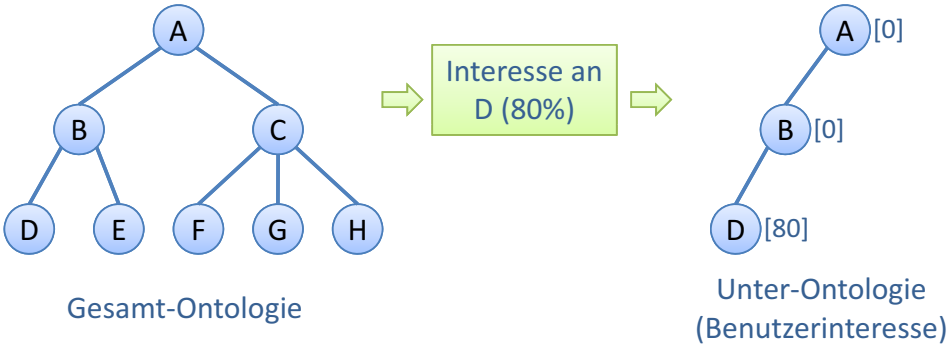


Abbildung 5.2: Erstellung einer Unter-Ontologie

Bei einer Änderung an einem Benutzer-Interesse, wird zwischen Hinzufügen eines neuen und Ändern eines bestehenden Interesses unterschieden. Wenn es sich um eine Änderung eines bestehenden Interesses handelt, wird der Interessenswert geändert und ggf. die Unter-Ontologie reduziert.

Abbildung 5.3 zeigt das Hinzufügen eines neuen Interesses, das sich auf die Gesamt-Ontologie von Abbildung 5.2 bezieht. Die Bestehende Unter-Ontologie wird um den fehlenden Teilbaum erweitert und der Interessenswert von 90 Prozent wird gesetzt.

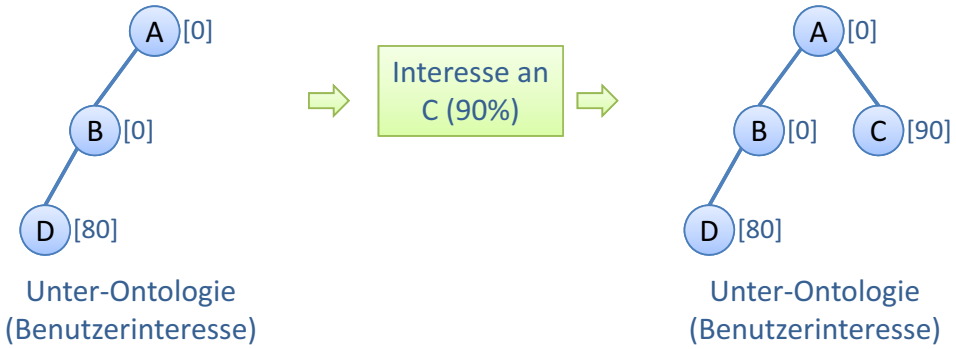


Abbildung 5.3: Erweitern einer Unter-Ontologie

Abbildung 5.4 visualisiert das Ändern eines bestehenden Interesses. Der Wert der bereits enthaltene Klasse C wird in diesem Beispiel von 90 Prozent auf 40 Prozent verändert.

Abbildung 5.5 zeigt die Reduktion einer Unter-Ontologie. Dies geschieht, wenn der Benutzer den Wert eines bestehenden Interesses auf 0 Prozent ändert. Ein Interessenswert von 0 Prozent kommt einem Entfernen des Interesses gleich. Somit wird zunächst der Wert von 80 Prozent auf 0 Prozent verändert und anschließend eine Reduktion des Baumes vorgenommen. Bei dieser Reduktion werden alle Klassen entfernt, die einen Interessenswert von 0 Prozent haben und keine Unterklassen besitzen.

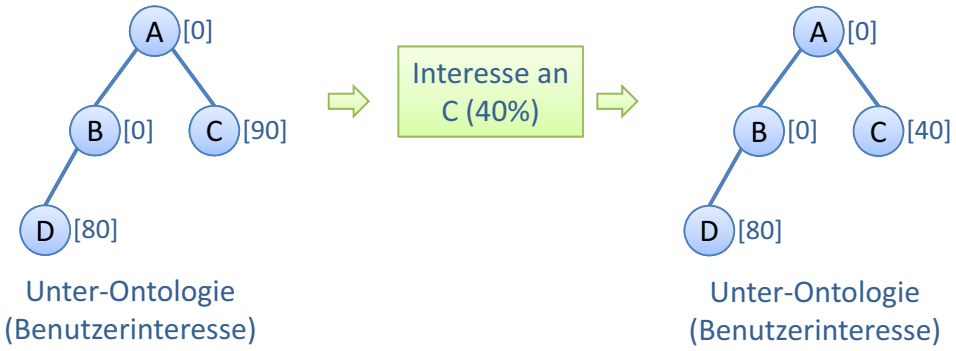


Abbildung 5.4: Ändern einer Unter-Ontologie

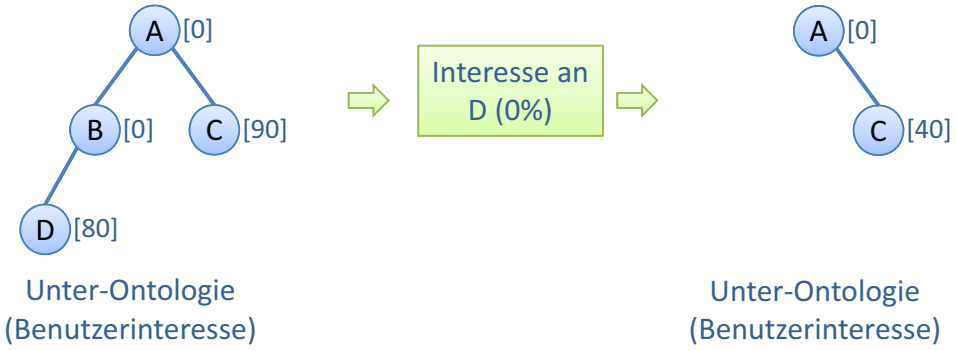


Abbildung 5.5: Reduktion einer Unter-Ontologie

Eine Unter-Ontologie ist ebenfalls eine `OWLOntologie`-Klasse und wird für die Übermittlung an den Client in eine `Interest`-Klasse konvertiert. Dies ermöglicht eine einfache Unterscheidung zwischen Gesamt-Ontologien (`Ontology`-Klasse) und Unter-Ontologien (`Interest`-Klasse). Die Klasse `Interest` besteht genau wie die Klasse `Ontology` aus dem Namen der zugehörigen Ontologie, dem Namen der Klasse und den zugehörigen Ober- bzw. Unterklassen. Zusätzlich besitzt sie noch ein einen Interessenwert (`interestValue`). Durch diese Struktur kann der Client die Unter-Ontologie entsprechend visualisieren und es ist möglich auf den jeweiligen Wert des Interesses zuzugreifen und ihn zu verändern.

MessageManager-Komponente

Die Klasse `MessageManager` beinhaltet die Klasse `Valuator`. Sie nimmt alle Anfragen die Mitteilungen betreffen entgegen.

In der `Valuator`-Klasse wird die Kategorisierung und Bewertung von Mitteilungen vorgenommen. Hierbei werden zwei Fälle unterschieden. Die Vorkategorisierung von Mitteilungen und die zeitgesteuerte Kategorisierung von Mitteilungen.

Die Vorkategorisierung ist als Rückmeldung für den Benutzer bei der Erstellung einer Mitteilung gedacht. (Siehe Anwendungsfall 3.4.1.1). Zunächst wird die erstellte (noch nicht veröffentlichte) Mitteilung indiziert und anschließend werden die Indizes der Mitteilung nach Übereinstim-

mungen mit den vorhandenen Gesamt-Ontologien durchsucht. Als Ergebnis wird eine Liste von Objekten der Klasse `FittingKategorie` zurückgegeben, die mit Zeichenketten des Textes korrelieren. Ein Objekt dieser Klasse enthält einen Kategorienamen (Oberklasse der betreffenden Ontologie), den Namen der Klasse und den bei der Kategorisierung ermittelten Treffer-Wert.

Die zeitgesteuerte Kategorisierung findet vor einem potentiellen Erhalt einer Mitteilung statt. (Siehe Anwendungsfall 3.4.1.2.) Zunächst werden die im System verfügbaren Mitteilungen indiziert. Bei der anschließenden Suche nach Übereinstimmungen werden die statischen und dynamischen Profileigenschaften eines Benutzers verwendet und die Ein- und Ausschlusslisten der jeweiligen Mitteilung berücksichtigt. Als Ergebnis wird eine Liste von Objekten der Klasse `FittingProfile` zurückgegeben, die mit Zeichenketten des Textes korrelieren. In dieser Klasse ist die betreffende Mitteilung und die übereinstimmenden dynamischen und statischen Eigenschaften enthalten. Bei den dynamischen Eigenschaften handelt es sich um Objekte der Klasse `Interest`, die den jeweilig ermittelten Treffer-Wert beinhaltet.

Indizierung: Für die Indizierung wird die javabasierte Indizierungs- und Suchbibliothek Apache Lucene[5] verwendet. Die Textinhalte werden nach Betreff und Inhalt der Mitteilung unterschieden und einem Analyzer (`MAStopAnalyzer`) übergeben. Dieser zerlegt die Daten nach definierten Regeln und bereitet sie einheitlich für den jeweiligen Index auf. Der `MAStopAnalyzer` eliminiert so genannte Stoppwörter wie z.B. „ein“, „einer“, „der“, „die“, „das“ und normalisiert die Zeichenketten in Kleinschreibung. Anschließend werden die Indizierung durchgeführt und die Suchräume aufbereitet.

Überprüfung der Gesamt-Ontologien: Die Überprüfung der Gesamt-Ontologien findet bei der Vorkategorisierung von Mitteilungen statt. Hier werden die Indizes der Mitteilung nach Übereinstimmungen mit den Gesamt-Ontologien des Systems durchsucht. Wenn eine Übereinstimmung mit einem Index festgestellt wird, gibt Apache Lucene einen Trefferwert zurück. Dieser Wert wird je nach Index (Betreff oder Inhalt der Mitteilung) unterschiedlich gewertet. Wenn ein Trefferwert für den Betreff einer Mitteilung ermittelt wird, wird dieser mit 100 Prozent gewichtet. Ein Trefferwert für den Inhalt einer Mitteilung wird mit 50 Prozent gewichtet. Anschließend werden beide Werte addiert und ganzzahlig in Prozent aufbereitet, wobei der daraus resultierende Wert 100 Prozent nicht überschreiten darf. Die entsprechende Klasse wird mit diesem Wert in der Klasse `FittingKategorie` gespeichert.

Überprüfung des statischen Profils: Das statische Profil eines Benutzers wird bei der zeitgesteuerten Kategorisierung von Mitteilungen einbezogen. Die Indizes der im System verfügbaren Mitteilungen werden nach Übereinstimmungen mit den vorhandenen statischen Angaben des Benutzers abgeglichen. Hierbei wird lediglich auf Vorhandensein der Eigenschaft geprüft. Dies geschieht zuerst für den Betreff-Index. Wenn hier ein Treffer gefunden wurde, wird der Index des Inhalts nicht überprüft. Der Inhalt der Mitteilung wird nur überprüft, wenn kein Treffer im Betreff der Mitteilung vorhanden ist. Sobald in einem der Indizes ein Treffer festgestellt worden ist, wird die entsprechende statische

[5] In dieser Arbeit wird Apache Lucene 2.2 verwendet. Siehe Kapitel 5.2.

Eigenschaft in die Liste der statischen Übereinstimmungen des `FittingProfile`-Objektes übernommen.

Überprüfung des dynamischen Profils: Diese Überprüfung findet bei der zeitgesteuerten Kategorisierung von Mitteilungen statt. Bei der Suche nach Übereinstimmungen mit dem dynamischen Profil bzw. den Interessen eines Benutzers werden die Indizes mit den Unter-Ontologien des Benutzers abgeglichen. Hierbei werden die Trefferwerte der beiden Indizes genau wie bei der Überprüfung der Gesamt-Ontologien unterschiedlich gewichtet (Betreff 100 Prozent und Inhalt 50 Prozent) und anschließend addiert. Die Höhe bzw. der Wert eines Interesses wird mit dem von Apache Lucene ermittelten Trefferwert abgeglichen. Wenn ein Benutzer z.B. an der Programmiersprache Java mit einem Wert von 80 Prozent interessiert ist und der Trefferwert der Suchbibliothek bei 0.1 (10 Prozent) liegt, würde keine Übereinstimmung festgestellt werden. Der Wert von 10 Prozent sagt aus, dass das Wort Java nur sehr selten in dem durchsuchten Text auftaucht und würde einem Interessenswert von mindestens 90 Prozent entsprechen. Die jeweiligen Ein- und Ausschlusslisten von Kategorien einer Mitteilung werden hierbei entsprechend berücksichtigt. Das heißt selbst wenn ein Wort sehr häufig in einem Text auftaucht, aber in der vom Erzeuger der Mitteilung erstellten Ausschlussliste vorhanden ist, wird dieses Wort übersprungen und es wird keine Übereinstimmung festgestellt. Bei einem Wort in der Einschlussliste einer Mitteilung wird selbst beim Abhandensein dieses Wortes eine Übereinstimmung festgestellt, sofern ein Interesse des jeweiligen Benutzers vorliegt. Das zutreffende bzw. den Vorgaben genügende Interesse wird in die Liste der dynamischen Übereinstimmungen des `FittingProfile`-Objektes übernommen.

Die Operationen wie das Laden und Ändern von Mitteilungen deligiert der `MessageManager` an den `PersistenceManager`.

SessionManager-Komponente

Die Klasse `SessionManager` ist für das Erzeugen und Überprüfen der Benutzer-Session zuständig. Sie enthält eine Liste aller gültigen Sessions. Bei der Anmeldung eines Benutzers wird eine neue Session erzeugt, die bei der Abmeldung wieder ungültig gemacht wird. Eine Benutzer-Session kann hier bei Bedarf auf Gültigkeit überprüft werden.

5.1.1.3 Data layer

Die Datenschicht (Data layer) ist für die Kommunikation mit der Datenbank zuständig. Zu ihr gehören die Klassen `PersistenceManager` und `Hibernate`. Die Hauptaufgabe dieser Schicht ist es, die definierten Objekte in der Datenbank zu speichern und aus ihr zu lesen. Hierbei wird durch die `Hibernate`-Komponente von der konkreten Datenbank abstrahiert.

PersistenceManager-Komponente

Die Klasse `PersistenceManager` kommuniziert mit der `Hibernate`-Klasse und holt sich aus deren `SessionFactory` die aktuelle Session. Mit Hilfe dieses `Session`-Objektes ist es möglich mit Objekten anstatt mit SQL-Statements zu operieren. Es ist das Bindeglied zwischen der Java-Applikation und den Hibernate-Diensten und bietet Methoden für Insert-, Update-, Delete- und Query-Operationen.

Die definierten Entitäten[6] können beispielsweise mit

```
session.load(Class cls, Serializable arg1)
```

aus der Datenbank ausgelesen und mit

```
session.save(Object obj)
```

in der Datenbank abgelegt werden.

Hierbei ist zu beachten, dass die `Session` nicht thread-save ist und möglichst mit kurzer Lebensdauer verwendet werden sollte. Innerhalb einer Session können mehrere Transaktionen nacheinander stattfinden. Konkurrierende Transaktionen sind nur in getrennten Sessions möglich.

Hibernate-Komponente

Die Hibernate-Komponente repräsentiert die in Abschnitt 4.1 dargestellte OR-Mapping-Komponente. Die Klasse `Hibernate` enthält die `SessionFactory`. Sie ist für das Laden der Konfiguration und die Mappings zuständig. Die `SessionFactory` ist thread-safe und kann für mehrere Threads verwendet werden.

Die Konfigurationsdatei (`hibernate.cfg.xml`) der `SessionFactory` enthält Informationen über den verwendeten JDBC-Treiber und den Ort, die Zugangsdaten und weitere Einstellungen der verwendeten Datenbank. Desweiteren enthält sie Mapping-Verweise auf die definierten Entitäten.

Ausschnitt der User-Entität (`user.hbm.xml`):

```
...
<hibernate-mapping package="de.masterthesis.client.model">
<class name="User" table="user" lazy="false">
<id name="id">
<generator class="native" />
</id>
<many-to-one name="account" column="accountId" />
<many-to-one name="contactInfo" column="contactInfoId"/>
<many-to-one name="profile" column="profileId" />
```

[6] Siehe Hibernate-Komponente.

```
<array name="receivedMessages" table="received">
<key column="userId" />
<index column="ind" />
<many-to-many column="messageId" class="Message" />
</array>

<array name="sendMessages" table="send">
<key column="userId" />
<index column="ind" />
<many-to-many column="messageId" class="Message" />
</array>

</class>
</hibernate-mapping>
```

Die User-Entität besitzt eine `Id`, einen `Account`, eine `ContactInfo`, ein `Profil` und hat Listen von empfangenen (`receivedMessage`) und gesendeten (`sendMessages`) Mitteilungen der Klasse `Message`.

Die Entitäten bilden die Grundlage für die entsprechenden Datenbanktabellen. Für jede Entität wird eine entsprechende Datenbanktabelle angelegt. Für die Verknüpfungstabellen (im obigen Beispiel `receivedMessages` und `sendMessages`) werden keine gesonderten Mapping-Einträge benötigt. Abbildung 5.6 zeigt das aus den Entitäten erzeugte Datenbankschema.

Für jede Entität existiert eine entsprechende Java-Klasse (POJO-JavaBean[7]). Diese Klassen bilden das Modell der Anwendung. (Siehe Abschnitt 5.1.3.)

5.1.1.4 Datenbank

Die Daten der Anwendung werden in einem relationalen Datenbankmanagementsystem (RDBMS) gespeichert.[8] Um eine objektorientierte Sicht auf die Tabellen und Beziehungen im RDBMS zu erhalten wird ein objektrelationaler Mapping-Service für Java verwendet. Dieses ojektrelationale Mapping wird mittels Hibernate[9] vorgenommen.

[7] POJO bedeutet „Plain Old Java Object“. Es handelt sich hierbei um ein „normales“ Java-Objekt.
[8] In dieser Arbeit wird MySQL 5.0.45 verwendet. Siehe Kapitel 5.2.
[9] In dieser Arbeit wird Hibernate 3.2.4 SP1 verwendet. Siehe Kapitel 5.2.

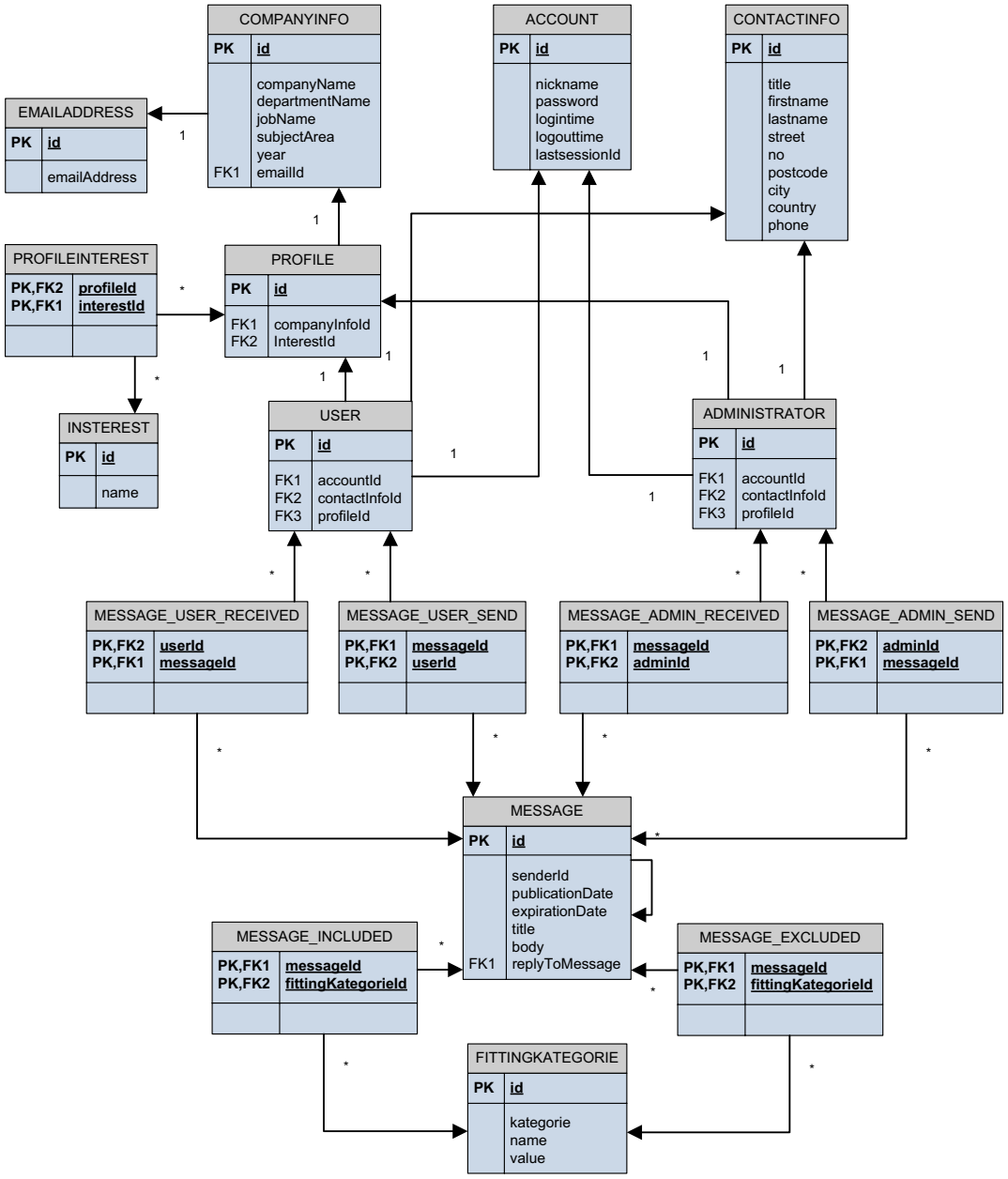


Abbildung 5.6: Entity Realationship-Diagramm

5.1.2 Client-Anwendung

Dieser Abschnitt erläutert die Komponenten der in Abschnitt 4.2 vorgestellten Client-Architektur. Die einzelnen Schichten werden durch die Abschnitte 5.1.2.1 für die Präsentationsschicht (Presentation layer), Abschnitt 5.1.2.2 für die Anwendungsschicht (Application layer) und Abschnitt 5.1.2.3 für die Dienstzugriffsschicht (Service access layer) beschrieben.

Abbildung 5.7 zeigt das Klassendiagramm (Komponenten) der Client Anwendung.

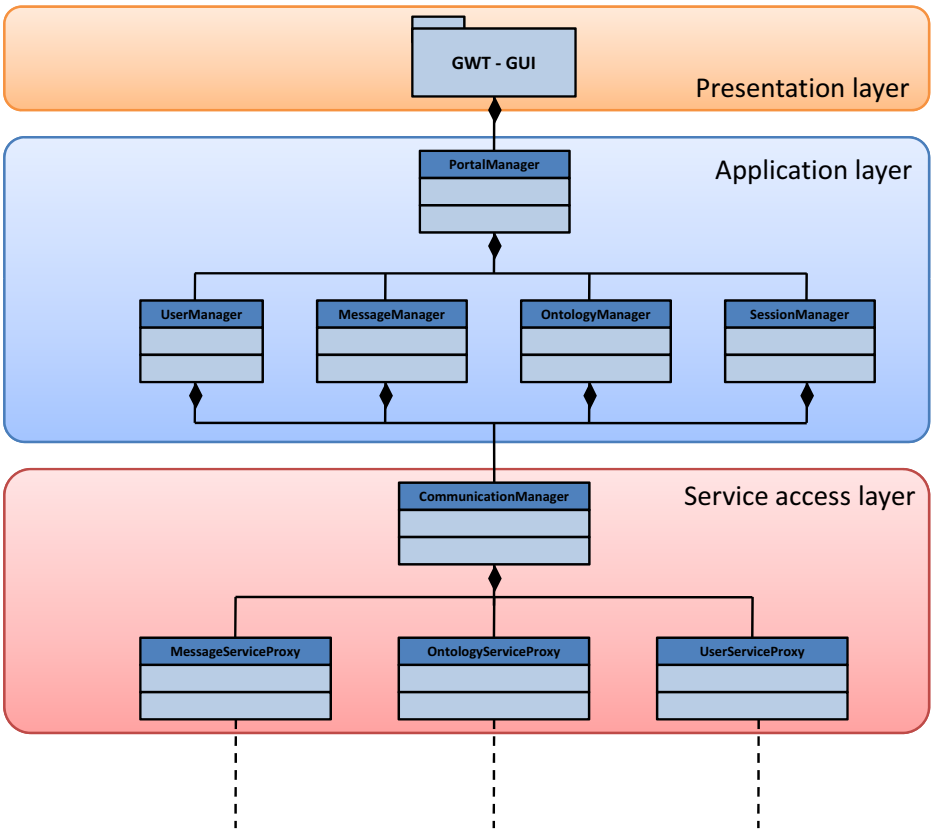


Abbildung 5.7: Klassendiagramm — Komponenten der Client Anwendung

5.1.2.1 Presentation layer

Die Präsentationsschicht (Presentation layer) visualisiert die Anwendung und bildet die Schnittstelle zwischen dem Benutzer und der Anwendung. Die GUI-Komponente `GWT-GUI` repräsentiert die verschiedenen Java-Klassen. Sie haben die Aufgabe die Entitäten der Anwendungsschicht (Application layer) darzustellen und Aktionen des Benutzers entgegen zu nehmen.

Um dem Benutzer eine intuitive Benutzung des Systems zu ermöglichen, wurden die Gestaltungsgrundsätze der Ergonomie bei der Mensch-System-Interaktion nach DIN EN ISO 9241-110:2006 [ISO06] berücksichtigt. Diese umfassen folgende Maßnahmen für die Dialoggestaltung:

- **Aufgabenangemessenheit:** „Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen."

  Die funktionalen Anforderungen aus Abschnitt 3.4 werden in einer geeigneter Form abgebildet. Der Benutzer soll hierdurch seine Aufgaben möglichst einfach erledigen können.

- **Selbstbeschreibungsfähigkeit:** „Ein Dialog ist in dem Maße selbstbeschreibungsfähig, in dem für den Benutzer zu jeder Zeit offensichtlich ist, in welchem Dialog, an welcher Stelle im Dialog sie sich befinden, welche Handlungen unternommen werden können und wie diese ausgeführt werden können."

  Durch die gewählte Menüstruktur erhält der Benutzer eine gute Übersicht über die verschiedenen Bereiche der Anwendung. Diese sind inhaltlich voneinander abgegrenzt und zeigen die zur Verfügung stehenden Handlungsmöglichkeiten auf. Das Menü ist dynamisch aufgebaut, so dass bei der Auswahl eines Bereiches nur dessen Handlungsmöglichkeiten zu sehen sind. Die anderen Bereiche werden dabei auf den Bereichsnamen reduziert.

- **Erwartungskonformität:** „Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z.B. seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie allgemein anerkannten Konventionen."

  Das Seitenlayout orientiert sich an der Studie von Michael Bernard, der die Erwartungshaltung von Personen für die Position von bestimmten Seitenelementen untersucht hat. In dieser Studie wurde z.B. nachgewiesen, dass die meisten Benutzer die Navigation auf der linken Seite erwarten [Ber02].
  Für die Visualisierung der Ontologien wurde eine Baumstruktur gewählt. Durch den semantischen Aufbau kann der Anwender von seiner eigenen Denkweise Gebrauch machen. Dies soll ihm beim Auffinden von Themengebieten und beim Hinzufügen und/oder Entfernen von Interessen unterstützen.

- **Lernförderlichkeit:** „Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet."

  Die Anwendung orientiert sich an der Metapher eines Schwarzen Brettes. Durch die Verwendung von Begriffen aus dem Alltag und durch entsprechende Bilder soll dem Benutzer das Verständnis erleichtert werden. Die Mitteilungen des Systems werden als „Aushänge" bezeichnet und der Menüpunkt besitzt ein Icon, das einen Zettel zeigt.

- **Steuerbarkeit:** „Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist."

  Durch die Verwendung des GWT bzw. der GWT-History-Klasse ist es möglich eine „Rückgängig-Funktion" zu implementieren. Somit hat der Benutzer die Möglichkeit die Aktivi-

täten des Systems zu beeinflussen und auch vorangegangene Schritte rückgängig zu machen.

- **Fehlertoleranz:** „Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann."

  Eingaben des Benutzers sollen zu keinen undefinierten Systemzuständen oder Systemabstürzen führen. Die Benutzereingaben werden überprüft und der Benutzer wird bei fehlerhaften Eingaben in geeigneter Weise benachrichtigt. Wenn der Benutzer z.B. bei der Registrierung einen bereits vorhandenen Benutzernamen auswählt, wird er hierüber mit einem roten Informationstext: „Der Benutzername ist bereits vergeben." direkt über dem Formular informiert. Der Cursor befindet sich anschließend im bereits markierten Eingabefeld. Bis auf das Passwort bleiben alle eingegebenen Informationen erhalten, so dass der Benutzer nur einen verfügbaren Benutzernamen und ein neues Passwort eingeben muss.

- **Individualisierbarkeit:** „Ein Dialog ist individualisierbar, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe sowie an die individuellen Fähigkeiten und Vorlieben des Benutzers zulässt."

  Die Anwendung basiert auf einem individuellen Benutzerprofil. Dieses kann von jedem Benutzer selbst angelegt werden. Hierbei hat er die Kontrolle, welche Interessen er eingibt und wie stark diese gewichtet werden sollen. Die Übersicht der Mitteilungen verändert sich entsprechend der Benutzereinstellungen.
  Da es sich um eine Webanwendung handelt kann die Darstellung der Seiten von vielen Browsern angepasst werden. Die sogenannte „Zoomfunktion" ermöglicht dem Benutzer die Inhalte für eine bessere Lesbarkeit anzupassen.[10]

Zur Darstellung dienen die im GWT Kapitel 2.3.1 angesprochenen Widgets. Die „GWT user-interface library" beinhaltet eine Reihe von Standards wie z.B. Knöpfe, Textfelder, Hyperlinks, Tabellen und Popup-Dialoge.[11] Zusätzliche Widget-Bibliotheken können eingebunden oder selbst implementiert werden.

Nach einer Eingabe des Benutzers bekommt dieser das Ergebnis angezeigt. Dies erfolgt wie in Abschnitt 2.3 dargestellt asynchron. Für die Realisierung wurde das MVC-Pattern (Model-View-Controller) eingesetzt. Eine entsprechende Anfrage wird der Anwendungsschicht übergeben, die die asynchrone Verarbeitung anstößt. Sobald die Verarbeitung abgeschlossen ist, wird die Präsentationsschicht informiert. Dies geschieht mit Hilfe des Observer-Patterns. Die Klasse, die sich für ein Ergebnis interessiert, registriert sich bei der `PortalManager`-Klasse der Anwendungsschicht, die alle Beobachter über aufgetretene Änderungen informiert. Durch dieses Vorgehen wird eine Trennung der Schichten realisiert, da das Model kein Wissen über das GUI hat.

[10] Für eine bessere Unterstützung dieser Funktion sind weitere Anpassungen notwendig.
[11] Für eine detaillierter Auflistung siehe [Goo07].

5.1.2.2 Application layer

Die Anwendungsschicht (Application layer) beinhaltet die clientseitige Anwendungslogik der Anwendung. Für jede der in Abschnitt 4.2.1 dargestellten Komponenten existiert eine Klasse, die entsprechend ihrer Verantwortlichkeit Operationen anbietet und den aktuellen Zustand der Anwendung enthält.

Die Klasse `PortalManager` ist der zentrale Zugriffspunkt für die Präsentationsschicht und erweitert die Observable-Klasse um die Präsentationsschicht über Änderungen zu informieren. Sie delegiert die Anfragen an die verantwortlichen Klassen und implementiert das Observer Interface um von diesen über das Ergebnis informiert zu werden.
Für eine kontinuierliche Aktualisierung des GUI wird ein Timer verwendet, der die Klassen `UserManager` und `MessageManager` in regelmäßigen Abständen benutzt. Hierbei wird die Methode `loadAllTopicalMessages()` (Abruf aller im System vorhandenen Mitteilungen) des MessageManagers und die Methode `loadInterestMessages()` (Abruf von Mitteilungen, die für den angemeldeten Benutzer von Interesse sind) des UserManagers aufgerufen. Durch diesen Mechanismus bleibt die Anzeige für den Benutzer aktuell und er kann über neu im System eingestellte und für ihn interessante Mitteilungen informiert werden.

UserManager-Komponente

Die Klasse `UserManager` ist für das Benutzermanagement zuständig. Sie nimmt alle Anfragen die einen Benutzer direkt betreffen entgegen.

Sie enthält den aktuell im System angemeldeten Benutzer. Hier kann der Status eines Benutzers abgefragt und erkannt werden, ob ein Benutzer angemeldet ist oder noch eine Registrierung aussteht. Alle Anfragen, die den Benutzer direkt betreffen, nimmt diese Klasse entgegen, die das Profil verwaltet. Änderungen am dynamischen und statischen Profil werden über diese Klasse abgehandelt. Die Mitteilungen, die einen Benutzer direkt betreffen, werden ebenfalls von dieser Klasse gehalten. Dies ist eine Liste der Klasse `FittingProfile`. Hierbei handelt es sich, wie bereits in Abschnitt 5.1.1.2 erwähnt, um Mitteilungen, die Übereinstimmungen mit den dynamischen und/oder statischen Eigenschaften des Benutzers aufweisen.

Anfragen, die eine Server-Aktivität benötigen, werden an die Klasse `CommunicationManager` weitergeleitet. Für die Benachrichtigung der PortalManager-Klasse und den Empfang von Ergebnissen erweitert diese Klasse die Observable-Klasse und implementiert das Observer-Interface.

MessageManager-Komponente

Die Klasse `MessageManager` ist für die Mitteilungsverwaltung zuständig. Hier werden alle Anfragen, die Mitteilungen betreffen, entgegen genommen.

Sie enthält eine Liste von `Message`-Objekten. Hierbei handelt es sich um alle aktuell im System zur Verfügung stehenden Mitteilungen. Diese werden über den `CommunicationMa-`

nager vom Server abgerufen und in dieser Klasse vorgehalten. Das Kategorisieren und Veröffentlichen von neuen Mitteilungen (siehe Anwendungsfall 3.4.1.1) sowie das Versenden einer Antwort (siehe Anwendungsfall 3.4.1.3) werden über diese Klasse angestoßen und an den CommunicationManager weitergeleitet. Für die Benachrichtigung der PortalManager-Klasse und den Empfang von Ergebnissen erweitert diese Klasse die Observable-Klasse und implementiert das Observer-Interface.

OntologyManager-Komponente

Die Klasse OntologyManager ist für die Verwaltung der Ontologien zuständig. Hier werden alle ontologiespezifischen Anfragen entgegen genommen.

Sie enthält eine Liste von Ontology-Objekten. Hierbei handelt es sich um alle im System verfügbaren Gesamt-Ontologien. Diese werden über den CommunicationManager vom Server abgerufen und in dieser Klasse vorgehalten. Um den PortalManager über Änderungen informieren zu können und selbst über Ergebnisse informiert werden zu können, erweitert sie die Observable-Klasse und implementiert das Observer-Interface.

SessionManager-Komponente

Die SessionManager-Klasse ist für die Verwaltung der Benutzer-Session zuständig.

Sie enthält die aktuelle Benutzer-Session (UserSession). Hier kann der Status der Session überprüft werden. Diese Klasse wird beim Anmelde- und Abmeldevorgang verwendet. Erst wenn bei einer Anmeldung eine gültige UserSession empfangen worden ist, kann der UserManager den Benutzer mit seinen Eigenschaften vom Server laden. Beim Abmelden wird die Session ungültig gemacht und der Benutzer wird durch den UserManager abgemeldet.

Die Anfragen, die eine Server-Aktivität benötigen, werden an die Klasse CommunicationManager weitergeleitet. Für die Benachrichtigung der PortalManager-Klasse und den Empfang von Ergebnissen erweitert diese Klasse die Observable-Klasse und implementiert das Observer-Interface.

5.1.2.3 Service access layer

Die Dienstzugriffsschicht (Service access layer) ist für die Kommunikation mit dem Server zuständig und bildet die Schnittstelle zwischen Client- und Server-Anwendung. Die einzelnen Klassen der Anwendungsschicht (Application layer) greifen auf die CommunicationManager-Klasse zu, um Daten an den Server zu senden bzw. um Daten vom Server zu erhalten.

Sie stellt die benötigten Methoden nach außen zur Verfügung und ruft die entsprechenden Dienste asynchron auf. Hierbei werden die generierten Service-Proxy-Klassen (UserServiceProxy, MessageServiceProxy und OntologyServiceProxy) initialisiert und der entsprechende Dienst wird mit einem Callback-Objekt aufgerufen. Nach dem Aufruf des Services erhält die CommunicationManager-Klasse eine Rückmeldung über Erfolg oder

Misserfolg. (Siehe Abschnitt 2.3.1). Die Rückmeldung wird entsprechend interpretiert und die Beobachter werden informiert.

Um die Klassen der Anwendungsschicht über Erfolg oder Misserfolg unterrichten zu können, erweitert die `CommunicationManager`-Klasse die Observable-Klasse.

5.1.3 Anwendungsmodell

Ausgehend von den funktionalen Anforderungen wurde ein UML-Modell ausgearbeitet. Abbildung 5.8 zeigt das UML-Klassendiagramm, das der Anwendung zugrunde liegt.

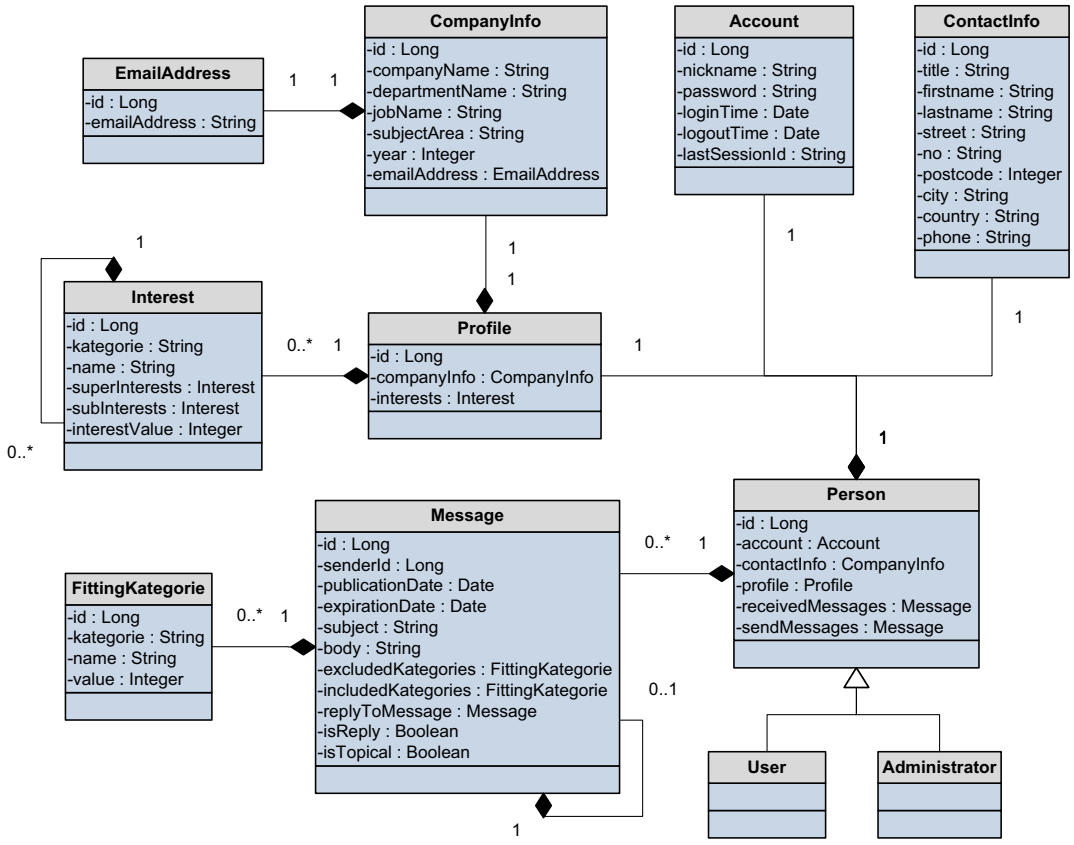


Abbildung 5.8: UML-Klassendiagramm

Person: Die Klasse `Person` ist eine Oberklasse der beiden Klassen `User` und `Administrator`. Die Klasse `User` entspricht dem in Abschnitt 3.4.1 beschriebenen Akteur. Die Klasse `Administrator` ist für zusätzliche, administrative Tätigkeiten vorgesehen. Eine `Person` ist durch eine künstliche Identifikationsnummer (ID) eindeutig identifizierbar. Sie besitzt ein Benutzer-Konto (`Account`), eine Kontakt-Information (`CompanyInfo`) und ein Benutzer-Profil (`Profile`). Zudem besitzt sie Listen von erhaltenen (`receivedMessages`) und gesendeten (`sendMessages`) Mitteilungen.

Account: Die Klasse `Account` ist ein Benutzer-Konto einer `Person`. Die Attribute `nickname` und `passwort` können bei der Registrierung vom Benutzer frei gewählt werden. Das Attribut `logintime` speichert das Datum der letzten Anmeldung und das Attribut `logouttime` das letzte Abmeldedatum. Das Attribut `lastSessionId` speichert die letzte Benutzer-Session. Hierdurch kann festgestellt werden, wann sich ein Benutzer zuletzt angemeldet hat, welche Session er verwendet hat und ob er sich korrekt vom System abgemeldet hat.

ContactInfo: Die Klasse `ContactInfo` beinhaltet alle persönlichen Kontaktinformationen eines Benutzers. Diese Daten werden für eine eventuelle Kontaktaufnahme mit dem Benutzer erhoben. Die Angabe dieser Daten ist bis auf den Vor- und Nachnamen freiwillig. Sie können direkt bei der Registrierung oder später bei der Verwendung des Systems angegeben werden. So kann ein Benutzer selbst entscheiden, ob und wann er diese Daten freigeben möchte. Durch diese Vorgehensweise werden alle in Kapitel 2.4.3.2 identifizierten Privatsphäre-Gruppen unterstützt. Sie können die Daten angeben, wenn genügend Vertrauen in das System vorhanden ist oder wenn sie für sich einen Nutzen identifiziert haben. (Siehe Kapitel 2.4.3.3.)

Profile: Die Klasse `Profile` entspricht dem Benutzerprofil, das für die Personalisierung innerhalb des Systems verwendet wird. (Siehe Kapitel 2.1.2.) Es besteht aus den Firmen-Informationen (`companyInfo`) und einer Liste von Interessen (`interests`). Hierbei entsprechen die Firmen-Informationen dem statischen Benutzerprofil und die Interessen dem dynamischen Benutzerprofil aus Abschnitt 3.4.1.

CompanyInfo: Die Klasse `CompanyInfo` beinhaltet alle statischen Informationen über einen Benutzer. Hierzu gehören der Firmenname `companyName`, der Fachbereich `departmentName`, die Tätigkeit `jobName`, die Fachrichtung `subjectArea` und die jeweilige Zugehörigkeitsdauer `year`. Desweiteren gehört auch eine E-Mail-Adresse `emailAddress` zu dem statischen Profil. Anhand dieser Informationen können aktuelle Mitteilungen von der Klasse `Valuator` auf Übereinstimmungen durchsucht werden (Siehe Abschnitt 5.1.1.2). Die E-Mail-Adresse kann vom Benutzer verwendet werden, um sich über neu veröffentlichte Mitteilungen informieren zu lassen.

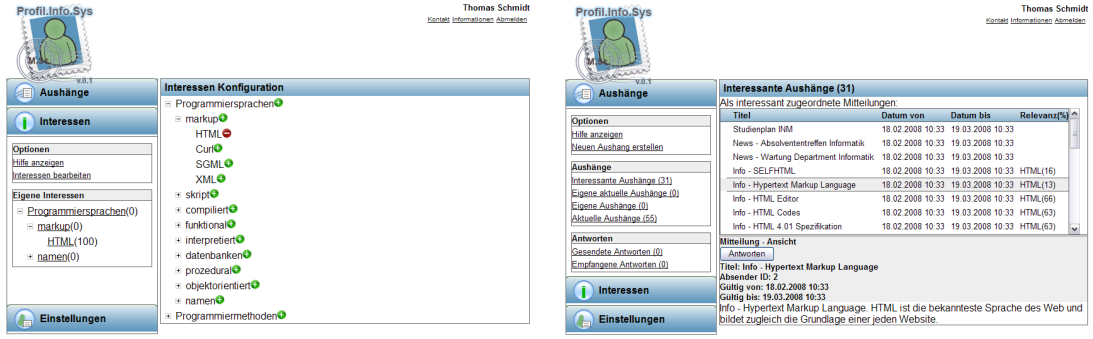
Interest: Die Klasse `Interest` entspricht dem dynamischen Benutzerprofil und steht für die Interessen eines Benutzers. Ein Benutzer kann, wie im Anwendungsfall 3.4.1.4 dargestellt, seine Interessen anhand von vorgegebenen Ontologien frei wählen und diesen einen Interessenwert zuweisen. Ein Interesse besteht aus einer Kategorie `kategorie`, die für die Oberklasse der Gesamt-Ontologie steht, dem eigenen Namen der Klasse `name` und den jeweiligen Ober- und Unterklassen `superInterest` und `subInterest`. Desweiteren enthält sie den vom Benutzer angegebenen Wert des Interesses `interestValue`. Der `Valuator` verwendet diese Klasse um Übereinstimmungen mit Mitteilungen zu identifizieren (siehe Abschnitt 5.1.1.2).

Message: Die Klasse `Message` steht für eine im System vorhandene Mitteilung. Sie beinhaltet die Identifikationsnummer des Erstellers bzw. Senders `senderId`, das Erstell- und Ablaufdatum `publicationDate` und `expirationDate` und den Betreff `subject`

und Inhalt `body` der Mitteilung. Desweiteren besitzt sie zwei Listen von Kategorien `FittingKategories`, die vom Benutzer bei der Veröffentlichung der Mitteilung gefüllt werden können. Hierbei handelt es sich um explizit eingeschlossene Kategorien `includedKategorie` und explizit ausgeschlossene Kategorien `excludedKategories`. Wenn eine Mitteilung einen Eintrag in diesen Listen enthält, wird dies von der `Valuator` Klasse entsprechend berücksichtigt (siehe Abschnitt 5.1.1.2). Das Attribut `isReply` gibt an, ob die Mitteilung eine Antwort auf eine Mitteilung `replyToMessage` ist. Das Attribut `isTopical` gibt an, ob das aktuelle Datum innerhalb des Gültigkeitsintervalls (Veröffentlichungsdatum – Ablaufdatum) liegt.

5.1.4 Fazit

Ein Ziel der Arbeit war eine prototypische Implementierung der oben beschriebenen Anwendung. Als Ergebnis ist eine Webanwendung entstanden, die sämtlichen funktionalen Anforderungen entspricht. Abbildung 5.9(a) zeigt einen Bildschirmausdruck der Interessenauswahl, bei der ein Interesse aus einer Liste von Ontologien ausgewählt werden kann. Abbildung 5.9(b) zeigt den Benutzerbildschirm bei der Anzeige von Mitteilungen, die dem Benutzer aufgrund seines Profils zugeordnet worden sind.



(a) Interessenauswahl (b) Mitteilung anzeigen

Abbildung 5.9: Bildschirmdruck des Prototyps

Der Quelltext der Anwendung ist auf der im Anhang A.3 beiliegenden DVD gespeichert.

5.2 Verwendete Software

Für die Entwicklung des Prototyps wurde die Programmiersprache Java (JRE 1.6.0_02) eingesetzt und folgende Software verwendet:

Apache Lucene: Für die Indizierung wurde die javabasierte Indizierungs- und Suchbibliothek Apache Lucene Version 2.2 verwendet.

Apache Tomcat: Für die Entwicklung der Webanwendung wurde der HTTP-Server von Apache Tomcat Version 5.5.25 verwendet.

djUnit: Zur Erstellung des Coverage-Reports wurde das Eclipse-Plugin djUnit Version 0.8.2 verwendet.

Eclipse: Für die Implementierung des Prototypen wurde Eclipse SDK Version 3.3.0 (Europa) verwendet.

Google Web Toolkit: Als Framework für die Entwicklung der Webanwendung wurde das Google Web Toolkit (GWT) Version 1.4.59 verwendet.

Hibernate: Für das objektrelationale Mapping wurde das Hibernate Framework Version 3.2.4 SP1 verwendet.

MySQL: Als Datenbank wurde MySQL Version 5.0.41 eingesetzt.

Pellet: Für Schlussfolgerungen innerhalb der verwendeten OWL-Ontologien wurde der Pellet - OWL DL Reasoner Version 1.5.0 eingesetzt.

Protégé: Zur Erstellung der Gesamt-Ontologien wurde der Ontologie-Editor Protégé Version 3.3.1 (build 430) verwendet. Zur Visualisierung der Ontologien wurde das GraphViz Plugin eingesetzt.

OWL API: Zur Verarbeitung der OWL-Ontologien wurde OWL API Version 2.1.1 verwendet.

5.3 Qualitätssicherung

Die Anwendung wurde bei der Implementierung für die Qualitätssicherung durch Modultests (Unit-Tests) überprüft. Hierfür wurden die Testfälle für die einzelnen Module parallel zur Entwicklung erstellt und bei Änderungen an der Anwendung wurden diese als Regressionstests durchgeführt.

Für die Testabdeckung der Anwendung wurde ein Coverage-Report mit Hilfe des djUnit-Plugins erstellt.[12] Hierbei bilden die Unit-Tests die Grundlage für die Testabdeckung.

[12] In dieser Arbeit wurde djUnit Version 0.8.2 verwendet. Siehe Kaptiel 5.2.

6 Systemevaluierung

In diesem Kapitel wird die Evaluation des entwickelten Prototypen diskutiert. In Abschnitt 6.1 wurden Untersuchungsziele und Erwartungen definiert. Als Untersuchungsmethode wurden Usability-Tests [Shn02] gewählt. Die Durchführung der Tests wird in Abschnitt 6.2 beschrieben. Abschnitt 6.3 diskutiert die Ergebnisse und bewertet die Zielerreichung.

6.1 Untersuchungsziele

In diesem Abschnitt werden die Versuchsziele definiert. Diese spiegeln die zu untersuchenden Werte wider und dienen als Entscheidungs- und Bewertungsgrundlage für die durchgeführten Tests.

Die Untersuchung soll Erkenntnisse darüber liefern, ob die Kombination eines offenen Benutzerprofils mit einer ontologiebasierten Navigation leicht von Benutzern anwendbar ist und einen Nutzen liefert. Hierfür soll überprüft werden, ob die Darstellung des Profils für den Benutzer transparent ist und er die Navigation versteht. Ein weiterer Fokus liegt auf dem Wohlempfinden des Benutzers. Hierbei gilt es den Gesamteindruck des Probanden zu ermitteln, um Rückschlüsse auf das Vertrauen in die Anwendung und eine spätere Verwendungsmöglichkeit machen zu können.

6.1.1 Erwartungen

Durch die klar gegliederten Strukturen der Webseite finden sich die Probanden schnell zurecht.

Durch die Verwendung von Ontologien und deren semantischen Aufbau fällt den Probanden die Navigation und Einstellung des Profils leicht. Durch die wiederkehrenden Strukturen verhält sich das System konform zu den Erwartungen des Benutzers. Dies festigt sein Verständnis der Anwendung und erleichtert ihm das Arbeiten.

Die Einstellungsmöglichkeiten anhand des Interessenswertes sind für die Probanden ungewohnt. Die Art der Eingrenzung von relevanten Mitteilungen könnte bei den Testpersonen zu Verständnisproblemen führen. (Ein eingestellter Interessenswert von 80 Prozent filtert alle Mitteilungen aus, die einen Relevanzwert von weniger als 20 Prozent haben.)

6.2 Untersuchungsdurchführung

Dieser Abschnitt beschreibt die Durchführung der Usability-Tests. In Abschnitt 6.2.1 wird der Umfang der Tests vermittelt. Anschließend wird in Abschnitt 6.2.2 der Ablauf beschrieben und ein Überblick über die verwendete Hardware und Software gegeben.

6.2.1 Umfang des Tests

Aus einer Studie von Nielsen und Landauer geht hervor, dass mehr als die Hälfte der Probleme einer Website mit drei Testpersonen gefunden werden. Mit einer Gruppe von fünf Personen erhält man die besten Ergebnisse [Nie00].

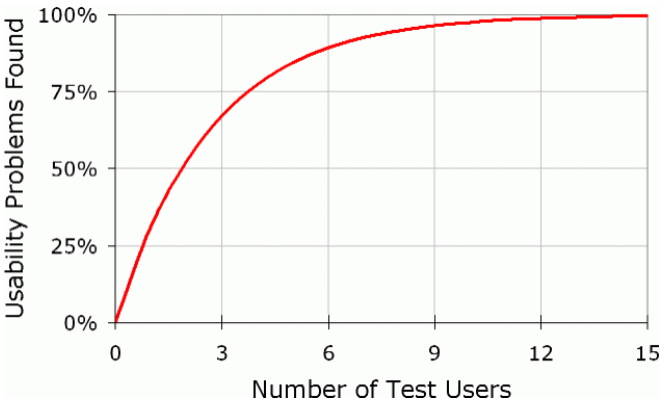


Abbildung 6.1: Problemfindungskurve von Nielsen und Landauer

Abbildung 6.1 zeigt die Problemfindungskurve von Nielsen und Landauer. Die zugrunde liegende Formel lautet: $N(1-(1-L)^n)$ Hierbei steht N für die Gesamtsumme der Probleme. Die Anzahl der Nutzer ist n und L ist die proportionale Anzahl der Problemen, die mit Hilfe einer Person gefunden werden können. Der Wert für L liegt bei 31 Prozent [Nie00].

Diese Formel wird in der Literatur mit unterschiedlichen Sichtweisen diskutiert. Einige Autoren, wie z.B. Perfetti und Landesman empfehlen mehr als 8 Personen um Usability Probleme zu finden [PL02]. Woolrych und Cockton haben in einer Studie die Formel von Nielsen und Landauer heuristisch evaluiert. Aus dieser geht hervor, dass 3 oder 5 Testpersonen angemessen sind, sofern diese nicht stark voneinander abweichen [WC01].

Insgesamt wurden sieben Probanden für den Usability-Test des Prototyps ausgewählt. Diese Anzahl scheint für diese Arbeit insofern als angemessen, da es sich bei den Testpersonen um Studenten der HAW-Hamburg handelt. Sie repräsentieren die Zielgruppe (vergleiche hierzu Abschnitt 3.3.2) und arbeiten in annähernd den selben Tätigkeitsgebieten. Die Bedingung zur Versuchsteilnahme war, dass keiner der Probanden den Prototyp vor dem Test gesehen bzw. benutzt hatte und dass die Testpersonen mit dem Umgang von Internetanwendungen vertraut sind.

6.2.2 Ablauf der Tests

Der Versuchsleiter empfing die Probanden und informierte sie mündlich, dass es sich um ein Experiment zur Mitteilungsvermittlung handelt, bei der ein Benutzerprofil die Grundlage bildet. Es wurde explizit darauf hingewiesen, dass nicht die persönliche Leistung des Teilnehmers im Mittelpunkt steht, sondern das Programm. Den Teilnehmern wurde erläutert, dass der Test für spätere Auswertungen aufgezeichnet werden sollte. Weiterhin wurden sie zum lauten Denken aufgefordert. Es folgte ein Überblick über die zu erledigenden Aufgaben.

Den Testpersonen wurden typische Aufgaben gestellt, die sie später in ähnlicher Form mit diesem Produkt erledigen würden.

Die funktionalen Anforderungen (Abschnitt 3.4) konnten durch entsprechende Aufgaben überprüft werden. Der Proband sollte sich als neuer Benutzer am System registrieren und ein Interessenprofil erstellen (Anwendungsfall 3.4.1.4). Durch diesen Vorgang wurden Mitteilungen vom System zugestellt (Anwendungsfall 3.4.1.2), die interpretiert werden sollten. Anschließend sollte die Testperson selbst eine Mitteilung im System veröffentlichen und die automatisch zugeordneten Kategorien interpretieren und bearbeiten (Anwendungsfall 3.4.1.1). Der Proband sollte die Anzahl der relevanten Mitteilungen mit Hilfe des Profils eingrenzen und auf eine frei gewählte Mitteilung antworten (Anwendungsfall 3.4.1.3).

Im Anschluss an die Aufgaben wurden den Testpersonen einige abschließende Fragen zum allgemeinen Umgang und zur Navigation des Systems gestellt. Zudem wurden Fragen zu einer prinzipiellen Praxistauglichkeit der Software gestellt und die Probanden wurden angeregt über zusätzliche Funktionalität und Themengebiete nachzudenken.

Der Test wurde vom Versuchsleiter moderiert, wobei darauf geachtet wurde, möglichst wenig Einfluss auf die Probanden zu nehmen. Dieses Vorgehen sollte die Versuchsdurchführung erleichtern, da es sich bei dem Prototyp um eine frühe Version handelt, bei der die Fehlertoleranz noch nicht vollständig implementiert worden war. An einigen Stellen im System gab es bei Benutzeraktionen noch keine expliziten Rückmeldungen über Erfolg eines Knopfdrucks. An diesen Stellen wurde der Proband vom Versuchsleiter darauf hingewiesen.

6.2.3 Verwendete Hardware

Für die Durchführung der Usability-Tests wurde folgende Hardware verwendet:

Testrechner: Als Testrechner für die Usability-Tests wurde ein Dell Optiplex 740 mit einem AMD Athlon 64 X2 Dual Core Prozessor 5200+ und 2 GB RAM verwendet.

Schnittrechner: Als Schnittrechner wurde ein HP XW 44000 Intel Core 2 6600 PC mit 2 GB RAM verwendet.

Videoaufzeichnung: Zur Videoaufzeichnung wurden insgesamt sechs Sony-Kameras verwendet. Vier dieser Kameras wurden in den Ecken des Raumes positioniert und zwei Kameras vor der Testperson. Durch diese Verteilung konnte alle Blickwinkel und das Gesicht des Probanden aufgenommen werden. Die Aufzeichnung der sechs Kamerasignale erfolgte auf zwei separaten Pentium D 3,4 GHz Terra PCs mit 2 GB RAM und jeweils vier Capture-Karten für die PAL-Eingangssignale.

Eyetracking: Für das Eyetracking wurde ein Tobii Eye Tracker (Modell X120) verwendet.

6.2.4 Verwendete Software

Für die Durchführung der Usability-Tests wurde folgende Software verwendet:

Apache Tomcat: Für den Anwendungstest der Webanwendung wurde der HTTP-Server von Apache Tomcat Version 5.5.25 verwendet.

MySQL: Als Datenbank wurde MySQL Version 5.0.41 eingesetzt.

Internet Explorer 7: Die Usability-Untersuchung wurde mit dem Microsoft Internet Explorer Version 7.0 durchgeführt.

Tobii Studio: Für die Eyetracking-Aufnahme auf dem Testrechner wurde Tobii Studio Version 1.1.12 eingesetzt. Auf dem Remote-Rechner im Regieraum wurde Tobii Studio Logger Version 1.1.12 verwendet.

Profil.Info.Sys: Der in dieser Arbeit entwickelte Prototyp befindet sich noch im Entwicklungsstatus. In den Usability-Tests wurde Profil.Info.Sys Version 0.1 untersucht.

6.3 Untersuchungsergebnisse

In diesem Abschnitt werden die Untersuchungsergebnisse diskutiert. Abschnitt 6.3.1 beschreibt die Ergebnisse zu den einzelnen Anwendungsfällen. Abschnitt 6.3.2 gibt eine Zusammenfassung der Ergebnisse.

6.3.1 Anwendungsfälle

Dieser Abschnitt beschreibt die Ergebnisse der Usability-Untersuchung zu den einzelnen Anwendungsfällen aus Abschnitt 3.4.1. Es wird diskutiert, ob die jeweilige Darstellungsform verstanden und intuitv verwendet wurde.

6.3.1.1 Mitteilung erhalten

Bei dem Erhalt von Mitteilungen (Anwendungsfall 3.4.1.2) war zu beobachten, dass die Darstellungsform angemessen ist. Die Probanden hatten keine Schwierigkeiten zu erkennen, weshalb das System ihnen Mitteilungen als interessant zugeordnet hat. Sie waren in der Lage Rückschlüsse auf ihre statischen und dynamischen Profileinstellungen zu ziehen.

6.3.1.2 Mitteilung beantworten

Die Möglichkeit auf angezeigte Mitteilungen zu antworten (Anwendungsfall 3.4.1.3) wurde von allen Probanden erkannt und verstanden. Die Darstellungsform stellte sich als intuitiv und erwartungskonform heraus.

6.3.1.3 Mitteilung verfassen

Das Veröffentlichen einer Mitteilung (Anwendungsfall 3.4.1.1) stellte ebenfalls kein Problem für die Testpersonen dar. Alle Probanden konnten die Mitteilung intuitiv erstellen. Bei der Kategorieüberprüfung wurde deutlich, dass ein Verständnis für das System vorhanden war, da die Testpersonen den Zusammenhang zwischen den Kategorien und dem eingegebenen Text erkennen konnten. Die Aufgabe zur Bearbeitung der Kategorien (Entfernen einer Kategorie) wurde schnell gelöst. Die Probanden sollten hierbei innerhalb der bereits bekannten Ontologien die entsprechende Kategorie auffinden. Hierbei stellte sich heraus, dass die Mehrheit der Probanden eine direkte Entfernungsmöglichkeit erwarten würde.

6.3.1.4 Profil bearbeiten

Um die Bearbeitung des Benutzerprofils (Anwendungsfall 3.4.1.4) zu überprüfen wurde den Probanden Aufgaben zum Erstellen eines eigenen Profils (Interessen auswählen) und zum Ändern eines bestehenden Profils (Interessen bearbeiten) gestellt.

Interessen auswählen

Bei der Navigation anhand der Ontologien bzw. innerhalb der Ontologien traten keine Probleme auf. Die Baumstruktur hat sich als geeignete Darstellung für die Ontologien erwiesen. Für die Testpersonen war dies eine bekannte Darstellungsform, die sich mit ihren Erwartungen deckte. Die Testpersonen konnten die gesuchten Themengebiete schnell finden. Es wurde beobachtet, dass die Probanden die Themengebiete auf unterschiedlichen Wegen zu ihrer Interessenliste hinzugefügt haben.

Zum Beispiel wurde die Programmiersprache Java sowohl über den Pfad

```
Programmiersprachen > Interpretiert > Java
```

als auch über

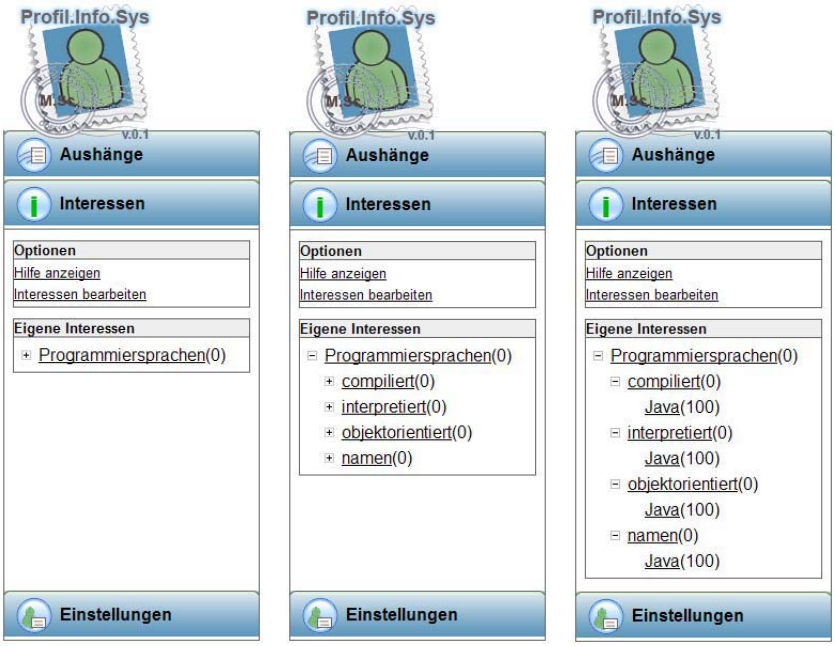
`Programmiersprachen > Objektorientiert > Java`

in das Profil übernommen.

Somit bestätigt sich die Erwartung, dass die Ontologie den Benutzer aufgrund des semantischen Aufbaus unterstützt.

Interessen bearbeiten

Für die Probanden war nicht direkt ersichtlich, wie sie den Interessenswert eines Themengebietes verändern konnten. Das Grundverständnis war vorhanden, aber die Funktion wurde an einer anderen Stelle erwartet. Die Darstellung des gesamten Ontologiebaumes mit den gewählten Interessenswerten stellte sich dabei als irreführend heraus.



(a) Interessenmenü Ebene 1 (b) Interessenmenü Ebene 2 (c) Interessenmenü Ebene 3

Abbildung 6.2: Interessenmenüs am Beispiel der Programmiersprache Java

In den Abbildungen 6.2(a), 6.2(b) und 6.2(c) ist zu sehen, dass der Interessenswert direkt hinter dem jeweiligen Interesse aufgeführt ist. Nachdem Java als Interesse ausgewählt wurde, wird die gesamte Unter-Ontologie im Menü dargestellt und der Wert für jeden Knoten aufgeführt. In diesem Beispiel hat Java einen Wert von 100 Prozent und die Oberinteressen einen Wert von 0 Prozent (Siehe Abschnitt 5.1.1.2). Dies wird durch einen Tooltip-Hilfetext und den Hilfe-Menüpunkt erläutert. Diese wurden von den Probanden nicht aufgerufen. Sie interpretierten den Wert als Anzahl der Interessen unterhalb des angezeigten Oberinteresses.

In diesem Beispiel ist zu sehen, dass Java sowohl unter kompiliert als auch unter interpretiert angeordnet ist. Dies wirkt zunächst wie ein Widerspruch, ist aber korrekt, da Java nicht direkt interpretiert wird. Der Java-Quell-Code wird durch den Java-Compiler in Bytecode kompiliert. Dieser ist binär und erst dann lauffähig, wenn er von der Java-Laufzeitumgebung interpretiert wird [Fla02].

Die Bearbeitung des eigenen Interessenswertes erfolgte mit Hilfe eines Schiebereglers, der den aktuell gewählten Interessenswert anzeigt. Sobald der Benutzer einen neuen Wert einstellt und speichert, aktualisiert das System die Übersicht über die als interessant zugeordneten Mitteilungen. Dieser Zusammenhang wurde von allen Probanden erkannt.

Die Tests lassen keine eindeutige Aussage zu, ob die gewählte Eingrenzungsfunktion geeignet ist. Die derzeitige Funktion filtert bei einem eingestellten Interessenwert von 80 Prozent alle Mitteilungen mit einem Relevanzwert von weniger als 20 Prozent heraus. Es gab sowohl Probanden, die diese Art der Eingrenzung erwartet hatten, als auch Probanden mit Verständnisproblemen dieser Darstellung. Diese Probanden erwarteten, dass ein eingestellter Wert von 80 Prozent nur noch Mitteilungen anzeigen würde, die eine Relevanz von mindestens 80 Prozent haben. Auch nach abschließender Befragung der Testpersonen konnte keine klare Präferenz festgestellt werden.

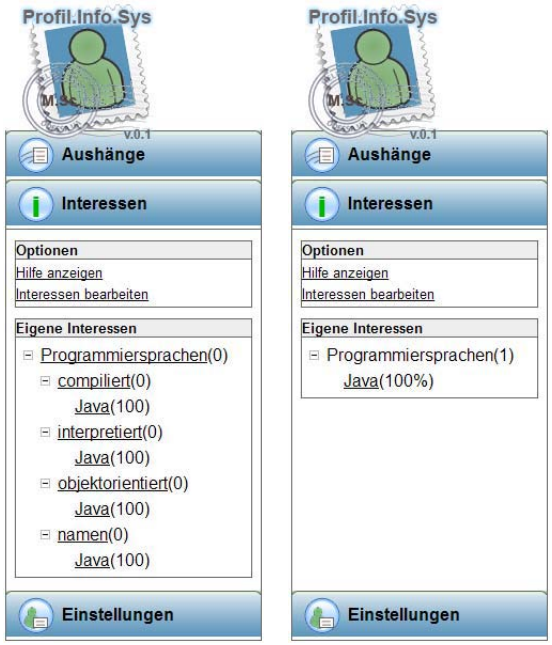
6.3.2 Fazit

Die Usability-Untersuchung des Prototyps des Profil.Info.Sys hat gezeigt, dass das System für die Studenten der HAW ein hilfreiches Instrument zur Informationsbeschaffung und -veröffentlichung ist.

Es ist für die Aufgabenerfüllung angemessen, da die Probanden in der Lage waren, die an sie gestellten Aufgaben zu bearbeiten. Die gewählten Funktionen und deren Darstellung sind für die jeweiligen Anwendungsfälle geeignet. Das Interessenmenü wurde aufgrund der Untersuchungsergebnisse überarbeitet und den Erwartungen der Probanden angepasst.

Abbildung 6.3(a) zeigt die vorherige Anzeige des Interessenmenüs und Abbildung 6.3(b) die geänderte Anzeige. Im vorherigen Menü wird die gesamte Unter-Ontologie der Interessenskategorie abgebildet und die einzelnen Interessenswerte angezeigt. Da die Testpersonen durch diese Darstellung irritiert wurden und die selbst gewählten Interessen nicht direkt auffinden konnten, wurde die Darstellung geändert. Die geänderte Anzeige zeigt nun die Oberkategorie mit der Anzahl der selbst gewählten Interessen. Als Unterpunkte stehen die entsprechenden Interessen mit ihrem aktuellen Interessenswert.

Es wird angenommen, dass die Benutzer durch diese Darstellung besser unterstützt werden. Die neue Anzeige ist deutlich kompakter und übersichtlicher. Allerdings wird in dem neuen Interessenmenü die Unter-Ontologie des Benutzers nicht mehr vollständig angezeigt. Um dem Benutzer eine Übersicht über diese Struktur geben zu können, wäre eine neue Anzeige erforderlich.



(a) Interessenmenü vorher (b) Interessenmenü nachher

Abbildung 6.3: Änderung des Interessenmenüs am Beispiel der Programmiersprache Java

Aus den Videoaufnahmen geht hervor, dass sich die Benutzer ein mentales Modell der Anwendung aufgebaut haben. Sie waren bis auf die oben angesprochene Anzeige nicht durch die transparente Darstellung und die Einstellungsmöglichkeiten des Benutzerprofils überfordert. Der Zusammenhang zwischen Benutzereingaben und den Ergebnissen bzw. der Rückmeldung des Systems war für die Teilnehmer ersichtlich. Die Navigation anhand von Ontologien hat die Probanden gut unterstützt (Siehe Abschnitt 6.3.1.4). Die sofortige Rückmeldung auf die getätigten Änderungen wurde positiv bewertet.

Die Videoaufnahmen der Usability-Tests sind auf der im Anhang A.3 beiliegenden DVD gespeichert.

7 Fazit

In diesem Kapitel wird in Abschnitt 7.1 eine Zusammenfassung der gesamten Arbeit gegeben. Die gewonnenen Ergebnisse der Arbeit werden in Abschnitt 7.2 bewertet. Abschnitt 7.3 gibt einen Ausblick über eine mögliche Weiterentwicklung.

7.1 Zusammenfassung

In dieser Arbeit wurde der Ansatz des ontologiebasierten, offenen Benutzerprofils „open user model“ mit ontologiebasierter Navigation für die Mitteilungsverarbeitung untersucht. Es wurden moderne Web-Technologien mit aktuellen Usability-Aspekten vereint um dem Benutzer einen komfortablen Zugang zu Mitteilungen und eine intuitive Nutzung des Systems zu ermöglichen.

In Kapitel 2 wurden die Grundlagen der relevanten Bereiche vorgestellt und deren wichtigsten Eigenschaften und Technologien erläutert.

Kapitel 3 beschreibt die Anforderungen an das entwickelte System. Zu Beginn wurde ein Ausgangsszenario vorgestellt, das dem Leser einen Überblick über die zu lösende Problemstellung gibt. Im Anschluss wurden Projekte, die an die Themenbereiche der Arbeit angrenzen vorgestellt und miteinander verglichen. Nach einer Diskussion über die Einflussfaktoren auf Ermittlungstechniken wurde die in dieser Arbeit verwendete Technik zum Ermitteln der Anforderungen an das System erläutert. Im Anschluss wurde der Kreis der befragten Personen vorgestellt und die verfolgten Ziele dargelegt. Die aus der resultierenden Umfrage (siehe Anhang A.2) gewonnenen Erkenntnisse wurden bei der Erstellung der funktionalen und nichtfunktionalen Anforderungen eingearbeitet.

In Kapitel 4 wurde die Software-Architektur des entwickelnden Systems unter Einbeziehung der herausgearbeiteten Anforderungen entworfen. Anschließend wurden die Komponenten und deren Zusammenspiel für die Server- und Client-Anwendung erläutert.

Kapitel 5 beschreibt die prototypische Implementierung der Anwendung und dessen Komponenten. Die resultierende Webanwendung entspricht den herausgearbeiteten Anforderungen. Nach der Übersicht über die bei der Implementierung verwendete Hard- und Software wurden die qualitätssichernden Maßnahmen vorgestellt.

In Kapitel 6 wurde die Evaluation des entwickelten Prototypen diskutiert, für die Usability-Untersuchungen gemacht worden sind. Zunächst wurden die Ziele und Erwartungen an die

Untersuchung identifiziert. Anschließend wurde der Umfang und der Ablauf der Untersuchungen in der Untersuchungsdurchführung vorgestellt. Die für die Durchführung der Tests verwendete Hard- und Software wurde vorgestellt, und die Untersuchungsergebnisse wurden anhand der Anwendungsfälle diskutiert. Hierbei konnte gezeigt werden, dass der Prototyp für die Aufgabenerfüllung angemessen ist und die Anforderungen erfüllt.

7.2 Bewertung

Ein Ziel dieser Arbeit war die Entwicklung eines Prototyps für eine ontologiebasierte Mitteilungsvermittlung und deren Evaluierung. Dieses Ziel kann als erreicht angesehen werden. Im Rahmen dieser Arbeit ist eine Webanwendung entstanden, die sowohl den funktionalen als auch den nichtfunktionalen Anforderungen entspricht. Das Anwendungsszenario am Beispiel der HAW kann generalisiert und auf Firmen oder Communities angewandt werden.

Durch den modularen Aufbau und die qualitätssichernden Maßnahmen kann die Anwendung einfach erweitert und gewartet werden. Sie ist plattformunabhängig und kann somit auf andere Umgebungen übertragen werden. Innerhalb der durchgeführten Tests zeigte sich die Anwendung als zuverlässig und hatte ein angemessenes Leistungsniveau. Aufgrund der geringen Menge an Testdaten und der begrenzten Untersuchungszeit kann dies jedoch nicht als repräsentativ angesehen werden. Die Usability-Untersuchungen belegen, dass die Anwendung leicht zu verstehen und intuitiv bedienbar ist.

Das System erfüllt die Metapher des Schwarzen Brettes und erweitert diese durch eine personalisierte Nutzung. Es bietet die Möglichkeit, Informationen für unbestimmte Empfänger innerhalb der Community zur Verfügung zu stellen. Die integrierte Antwortmöglichkeit erleichtert die Kontaktaufnahme zum Verfasser. Der Benutzer des Systems hat ein Profil, das er selbstständig verwalten kann. Anhand dieses Profils werden ihm Vorschläge für interessante Mitteilungen gemacht. In der vorliegenden Version des Prototyps ist die Einstellung nur manuell durch den Benutzer möglich. Leider war es im Rahmen dieser Arbeit nicht mehr möglich, Einstellungsmöglichkeiten durch Bewertungen oder durch das Verhalten eines Benutzers zu integrieren.

Durch die durchgeführten Usability-Untersuchungen wurde gezeigt, dass das System eine gute Übersicht über die vorhandenen Mitteilungen bietet. Die Benutzer empfanden den Umgang mit der Anwendung als angenehm und fanden sich schnell zurecht. Die Navigation anhand von Ontologien ist bei der Profilerstellung hilfreich, da sie verschiedenen Denkweisen der Benutzer unterstützt. Die Anwendung wurde so entworfen, dass der Nutzer selbst entscheiden kann welche persönlichen Daten er preisgibt. Seine gespeicherten Daten kann er jederzeit einsehen und anpassen. Zusammen mit dem geäußerten Wohlempfinden der Probanden gibt dies einen Hinweis darauf, dass das Vertrauen gestärkt werden könnte. Allerdings lässt dies keinen endgültigen Schluss auf eine tatsächliche Stärkung zu und müsste durch Langzeitstudien genauer überprüft werden.

Das Ziel eines komfortableren Zugangs zu Informationen kann als erreicht angesehen werden. Der Benutzer hat eine gute Übersicht über alle vorhandenen Mitteilungen und bekommt

Vorschläge für potentiell interessante Mitteilungen. Er muss die Quellen nicht eigenständig durchsuchen und wird durch die Ontologien in seiner Denkweise unterstützt.

7.3 Ausblick

Der entwickelte Prototyp ist bereits für die geforderte Aufgabenerfüllung einsetzbar. Es wurden Untersuchungen mit Testpersonen durchgeführt die sowohl positive Ergebnisse als auch Hinweise auf Erweiterungsmöglichkeiten liefern. Für eine zukünftige Version sind einige Weiterentwicklungen und Erweiterungen vorstellbar und notwendig, die nachfolgen beschrieben werden.

Für einen produktiven Einsatz müssen die vorhandenen Beispiel Ontologien durch geeignete Ontologien ersetzt werden. Diese können z.B. durch Befragungen der zukünftigen Nutzer des Systems ermittelt werden. Die Ergebnisse aus den Benutzerfragebögen sowie den Benutzerstudien dieser Arbeit geben hierbei erste Hinweise für das Anwendungsszenario. Es sind sowohl studienfachbezogene als auch studienfachübergreifende und allgemeine Themenbereiche vorstellbar. Das gesamte Spektrum der Lehrinhalte könnte für die jeweiligen Fachbereiche abgebildet werden. In studienfachübergreifenden Bereichen könnten Sonderveranstaltungen, Studienberatung oder organisatorische Themen wie Prüfungsordnungen oder Vorschriften angesiedelt werden. Allgemeine Themen wie etwa Programmiersprachen oder Programmiermethoden könnten in den allgemeinen Bereichen aufgeführt werden. Innerhalb der Ontologien ist es möglich, bereits bestehende Ontologien zu verwenden. Bislang bietet der Prototyp jedoch keine Möglichkeit, um einen Abgleich zwischen gleichen Themen herzustellen. Wenn z.B. in einer AI (Angewandte Informatik) Ontologie die Kategorie „Java“ ausgewählt wird, wird eine entsprechende AI Unter-Ontologie für den Benutzer erstellt. Wenn die AI-Ontologie hierbei auf Programmiersprachen Ontologie zugreift, wäre es aber sinnvoll zusätzlich eine Unter-Ontologie von Programmiersprachen mit dem Interesse „Java“ zu erstellen.

Ein Administrationsbereich für die Anwendung ist vorgesehen, aber derzeit noch nicht implementiert. Funktionen wie eine Rechteverwaltung von Benutzern oder eine Möglichkeit zur Erstellung und Bearbeitung von Ontologien sind hier vorstellbar.

Im Rahmen der Benutzerstudien konnten zusätzliche Funktionalitäten ermittelt werden, die den Benutzer im Umgang mit der Anwendung unterstützen könnten. Eine Suchfunktion anhand des Namens von Kategorien würde den Benutzer beim Auffinden von Interessen unterstützen. Er könnte schneller zu einem bestimmten Interesse navigieren und könnte durch die Ontologien sehen in welchen Bereichen es liegt. Eine Filterfunktion für Mitteilungen würde dem Benutzer ermöglichen eine bessere Übersicht zu erhalten. Momentan wird dem Benutzer eine Übersicht von allen vorhandenen Mitteilungen unter Berücksichtigung aller Interessen gegeben. Mit einer Filterfunktion könnte er diese eingrenzen und sich nur die Mitteilungen eines speziellen Interesses anzeigen lassen oder Mitteilungen von bestimmten Interessensbereichen gezielt ausblenden lassen.

In der aktuellen Version des Prototyps erfolgen die Anpassungen des dynamischen Benutzerprofils durch den Benutzer selbst. Es ist vorgesehen, dass das Profil nicht nur manuell sondern auch automatisch eingestellt werden kann. Hierbei kann zwischen halbautomatischen und vollautomatischen Anpassungen unterschieden werden. Halbautomatische Anpassungen erfordern eine Aktion des Benutzers. Mitteilungen können vom Benutzer direkt nach Interesse bzw. Relevanz gewertet werden. Hierbei sind mehrere Bewertungsskalen vorstellbar, wie z.B. eine einfache Bewertung von „interessant oder uninteressant“ oder feinere Bewertungen wie „1 = uninteressant, bis 10 = sehr interessant“. Je nach Bewertung verändert das System anschließend die jeweiligen Interessenswerte innerhalb des Profils. Vollautomatische Anpassungen erfordern keine Aktion des Benutzers. Das System würde das Verhalten eines Benutzers bewerten. So könnte das Leseverhalten eines Benutzers Rückschlüsse auf die Relevanz einer Mitteilung zulassen. Die Textlänge im Verhältnis zur Lesezeit könnte hierbei ein angemessener Indikator sein. Eine Antwort auf eine Mitteilung würde auf ein direktes Interesse hinweisen. Für solche automatischen Anpassungen sind geeignete Bewertungsformeln notwendig, die wiederum durch Benutzerstudien evaluiert werden sollten.

Da die Anwendung für Communities vorgesehen ist, ist eine Erweiterung durch „social navigation“-Komponenten denkbar.
Das System könnte dem Benutzer Vorschläge für eventuell interessante Themenbereiche aufgrund von Profilen anderer Mitglieder machen. Ein bekanntes Beispiel in diesem Bereich liefert die Firma Amazon: „Personen die dieses Buch gekauft haben, haben auch folgende Bücher gekauft“. Dies lässt sich im Anwendungsszenario der HAW z.B. auf den Fachbereich und das Fachsemester übertragen: „Studenten im gleichen Semester interessieren sich für folgende Bereiche“. Hierfür wäre ein Abgleich der bereits vorhandenen Profile der Mitglieder erforderlich. Es ist zu erwarten, dass hierdurch, nach einer gewissen Anlaufphase, ein Mehrwert für den Benutzer entsteht.
Desweiteren ist eine Kommentarfunktion für Mitteilungen denkbar, wie man sie aus Foren kennt. Der Benutzer könnte wählen, ob seine Antwort auf eine Mitteilung öffentlich sichtbar gemacht werden soll. Durch diese Funktion könnten die Mitglieder voneinander profitieren und der Verfasser der Mitteilung würde entlastet werden. So würde vermieden werden, dass die selbe Anfrage mehrmals beantwortet werden muss. Interessierte könnten an mehr Informationen gelangen.

Abbildungsverzeichnis

Literaturverzeichnis

[ABG+07] AHN, JAE WOOK, PETER BRUSILOVSKY, JONATHAN GRADY, DAQING HE und SUE YEON SYN: *Open user profiles for adaptive news systems: help or harm?* ACM, 2007.

[Acq04] ACQUISTI, ALESSANDRO: *Privacy in Electronic Commerce and the Economics of Immediate Gratification*. ACM Conference on Electronic Commerce, 2004.

[BC92] BELKIN, NICHOLAS J. und W. BRUCE CROFT: *Information Filtering and Information Retrieval: Two Sides of the Same Coin*. Communications of the ACM, 1992.

[Ben07] BENCHLEY, ROBERT: *FOAF Project - Friend of a Friend*, 2007. http://www.foaf-project.org/ - Abruf am 01.03.2008.

[Ber02] BERNARD, MICHAEL: *Examining User Expectations for the Location of Common E-Commerce Web Objects*. Software Usability Research Laboratory, 2002. http://psychology.wichita.edu/surl/usabilitynews/41/web_object-ecom.htm -.

[BKN07] BRUSILOVSKY, PETER, ALFRED KOBSA und WOLFGANG NEJDL: *The Adaptive Web: Methods and Strategies of Web Personalization*. Nummer 1. Springer Verlag Berlin, 2007. ISBN-10: 3540720782.

[BM04] BRICKLEY, DAN und LIBBY MILLER: *FOAF Vocabulary Specification*, 2004. http://xmlns.com/foaf/spec/ - Abruf am 01.03.2008.

[Bun83] BUNDESVERFASSUNGSGERICHT: *BVerfG - Volkszählungsurteil - Aktenzeichen: 1 BvR 209/83 u.a. vom 15.12.1983*, 12 1983. BVerfGE , Band 65, Seite 1 ff. http://www.datenschutz.de/themen/literatur/detail/?catchid=1051&litid=603 - Abruf am 01.03.2008.

[CCCJ04] CARREIRA, RICARDO, JAMIE M. CRATO, DANIEL CONCALVES und JOAQUIM A JORGE: *Evaluating Adaptive User Profiles for News Classification*. ACM, 2004.

[Com90] COMMISSION, DATA PROTECTION EUROPEAN: *United Nations guidelines concerning Computerized personal data files*, December 1990. http://ec.europa.eu/justice_home/fsj/privacy/instruments/un_en.htm - Abruf am 01.03.2008.

[CP07] CLARK und PARSIA: *Pellet: An OWL DL Reasoner*, 2007. http://pellet.owldl.com/ - Abruf am 01.03.2008.

[CR03] COOPER, ALAN und ROBERT REIMANN: *About Face 2.0 - The essentials of interaction design*. Wiley, 2003.

[Dör03] DÖRNER, JAN-HENDRIK: *Personalisierung im Internet - Persönliche Empfehlungen mit Collaborative Filtering*, Band 1. Dr. Kovac Verlag, 2003.

[Ede06] EDELMAN, BENJAMIN: *Adverse Selection in Online "Trust"Certifications*. Harvard University - working draft, 2006.

[Fla02] FLANAGAN, DAVID: *Java in a nutshell*, Band 4. O Reilly, 2002.

[Fog03] FOGG, B.J.: *Persuasive Technology: Using Computers to Change What We Think and Do*. Morgan Kaufmann, 2003.

[Gar03] GARSTKA, HANSJÜRGEN: *Bürgerrechte im Netz*, Band 382, Kapitel Informationelle Selbstbestimmung und Datenschutz: Das Recht auf Privatsphäre, Seiten 48–70. Bundeszentrale für politische Bildung, 2003.

[Gar05] GARRETT, JESSE JAMES: *Ajax: A New Approach to Web Applications*, 2 2005. http://adaptivepath.com/ideas/essays/archives/000385.php - Abruf am 01.03.2008.

[Goo07] GOOGLE: *Google Web Toolkit - Build AJAX apps in the Java language*, 2007. http://code.google.com/webtoolkit/ - Abruf am 01.03.2008.

[Goo08] GOOGLE: *Google Alerts (BETA)*, 2008. http://www.google.de/alerts - Abruf am 01.03.2008.

[GRG04] GRASL, OLIVER, JÜRGEN ROHR und TOBIAS GRASL: *Prozessorientiertes Projektmanagement*. Carl Hanser Verlag München, 2004.

[Gru93] GRUBER, THOMAS R.: *A Translation Approach to Portable Ontology Specifications*. Knowledge Systems Laboratory - Technical Report, 1993.

[HL04] HAUSER, TOBIAS und ULRICH M. LÖWER: *Web Services - Die Standards*. Galileo Computing, 2004.

[Hor04] HORRIDGE, MATTHEW: *OWLViz - A visualisation plugin for the Protégé OWL Plugin*. The University Of Manchester, 2004. http://www.co-ode.org/downloads/owlviz/ - Abruf am 01.03.2008.

[HTG06] HUI, KAI-LUNG, BERNARD C.Y. TAN und CHYAN-YEE GOH: *Online Information Disclosure: Motivators and Measurements*. ACM Transactions on Internet Technology, 2006.

[Inc07] INC., CHOICESTREAM: *Personalization Survey*. 2007. http://www.choicestream.com/surveyresults/ - Abruf am 01.03.2008.

[ISO06] ISO, INTERNATIONAL ORGANISATION FOR STANDARDISATZION: *Ergonomics of human-system interaction – Part 110: Dialogue principles*, 2006. www.iso.org.

[KC05a] KOBSA, ALFRED und LORRIE CRANOR: *Proceedings of the UM05 Workshop 'Privacy-Enhanced Personalisation'*. 2005. http://www.isr.uci.edu/pep05/papers/w9-proceedings.pdf - Abruf am 01.03.2008.

[KC05b] KUMARAGURU, PONNURANGAM und LORRIE FAITH CRANOR: *Privacy Indexes: A Survey of Westin's Studies*. Institute for Software Research International, Carnegie Mellon University and by the Carnegie Mellon CyLab, 2005.

[KCS06] KOBSA, ALFRED, RAMNATH K. CHELLAPPA und SARAH SPIEKERMANN: *Proceedings of the CHI2006 Workshop on Privacy-Enhanced Personalisation*. National Science Foundation, 2006.

[KH98] KATZ-HAAS, RAISSA: *Summary of Ten Guidelines for User-Centered Web Design*. Society for Technical Communication, 1998. http://www.stcsig.org/usability/topics/articles/ucd%20_web_devel.html - Abruf am 01.03.2008.

[Kob07] KOBSA, ALFRED: *Privacy-Enhanced Personalization*. Communications of the ACM, 50(8), 2007.

[KR02] KUROSE, JAMES F. und KEITH W. ROSS: *Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet*. Addison-Wesley, 2002.

[Kru06] KRUG, STEVE: *Don't make me think! Web Usability: Das intuitive Web*. Pearson Education Inc, 2 Auflage, 2006.

[May99] MAYHEW, DEBORAH J.: *The usability engineering Lifecycle*. Morgan Kaufmann, 1999.

[Nie00] NIELSEN, JAKOB: *Why You Only Need to Test With 5 Users*, 3 2000. http://www.useit.com/alertbox/20000319.html - Abruf am 01.03.2008.

[O'R05] O'REILY, TIM: *What Is Web 2.0*, 09 2005. http://www.doodle.ch/participation.html?pollId=f9wbmg4v4gr4eids - Abruf am 01.03.2008.

[PL02] PERFETTI, CHRISTINE und LORI LANDESMAN: *Eight is not enough*, 2002. http://www.uie.com/articles/eight_is_not_enough/ - Abruf am 18.03.2008.

[Pre08] PRESSWATCH: *PressWatch - Media Monitoring Services*, 2008. http://www.presswatch.de/de/ - Abruf am 01.03.2008.

[PS04] PARSIA, BIJAN und EVREN SIRIN: *Pellet: An OWL DL Reasoner*. MINDSWAP Research Group, 2004.

[RG02] RUPP, CHRIS und SOPHIST GROUP: *Requirements-Engineering und -Management - Professionelle, iterative Anforderungsanalyse für die Praxis*. Hanser, 2002.

[SC05] SEO-CONSULTING: *Eye-Tracking Studie zeigt Webseiten Blickfeld*, 2005. http://www.seo-consulting.de/pages/news-380.php - Abruf am 01.03.2008.

[Shn02] SHNEIDERMAN, BEN: *User Interface Design*. mitp-Verlag, Bonn, 2002.

[Sta07] STANFORD: *Ontologie Editor - Protégé*, 2007. http://protege.stanford.edu/ - Abruf am 01.03.2008.

[Ste06] STEFANER, MORITZ: *Stickees*, 2006. http://incom.org/code/projekte/projekt_anzeigen.php?4,135,17,0,0,156 - Abruf am 01.03.2008.

[T.G93] T.GRUBER: *What is an Ontology?*, 1993. http://www-ksl.stanford.edu/kst/what-is-an-ontology.html - Abruf am 01.03.2008.

[W3C04a] W3C, WORLD WIDE WEB CONSORTIUM: *OWL - Web Ontology Language*, 2004. http://www.w3.org/TR/owl-features/ - Abruf am 01.03.2008.

[W3C04b] W3C, WORLD WIDE WEB CONSORTIUM: *RDF - Resource Description Framework*, 2004. http://www.w3.org/TR/rdf-syntax-grammar/ - Abruf am 01.03.2008.

[W3C04c] W3C, WORLD WIDE WEB CONSORTIUM: *RDF Schema*, 2004. http://www.w3.org/TR/rdf-schema/ - Abruf am 01.03.2008.

[W3C06] W3C, WORLD WIDE WEB CONSORTIUM: *Notation 3*, 2006. http://www.w3.org/DesignIssues/Notation3.html - Abruf am 01.03.2008.

[WA91] WESTIN, ALAN F. und HARRIS LOUIS & ACCOCIATES: *Harris-Equifax Consumer Privacy Survey*. Tech. rep., 1991.

[WC01] WOOLRYCH, ALAN und GILBERT COCKTON: *Why and When Five Test Users aren't Enough*, 2001. http://www.netraker.com/nrinfo/research/FiveUsers.pdf - Abruf am 18.03.2008.

[Wes67] WESTIN, ALAN F.: *Privacy and freedom*. Atheneum, New York, 1 Auflage, 1967.

[Win93] WINSKEL, GLYNN: *The Formal Semantics of Programming Languages: An Introduction*. MIT Press, 1993.

Glossar

Ajax: Abkürzung für “Asynchronous JavaScript and XML”. Ajax ist ist ein Konzept zur Entwicklung von Web-Anwendungen.

API: Abkürzung für “Application Programming Interface”. Eine definierte Schnittstelle eines Programms, an die andere Programme angeschlossen werden können.

Collaborative Filtering: Ein Verfahren zur Personalisierung von Inhalten. Benutzer werden auf Ähnlichkeiten in ihren Benutzerprofilen hin untersucht, aus denen sich bestimmte Prognosen zur Individualisierung von Inhalten ableiten lassen.

Eye-Tracking: Eine Methode der Blickbewegungregistrierung. Speziell entwickelte Videokameras erfassen die Augen von Menschen und ermitteln die entsprechenden Blickbewegungspositionen.

GUI: Abkürzung für “Graphical User Interface”. Eine grafische Benutzeroberfläche, die dem Nutzer eines Computers eine Interaktion mit der Anwendung über grafische Symbole ermöglicht.

GWT: Abkürzung für “Google Web Toolkit”. Ein Java Framework für die Entwicklung von Web-Anwendungen.

HTTP: Abkürzung für “Hypertext Transfer Protocol”. Ein Protokoll der ISO-OSI-Anwendungsschicht zum Übertragen von Daten in einem Netzwerk.

IDE: Abkürzung für “Integrated Development Environment”. Ein Anwendungsprogramm für die Entwicklung von Software.

Information Filtering: Ein Verfahren für die Personalisierung von Inhalten. Eine Datenmenge wird durch die Verwendung von Benutzerdaten eingegrenzt.

Komponente: Eine (Software-)Komponenete ist ein Teil einer Software. Sie dient zur Kapselung von Funktionalitäten und ist über eine definierte Schnittstelle erreichbar.

Ontologie: In der Informatik bezeichnet eine Ontologie eine konzeptuelle Formalisierung von Wissensbereichen und Begriffssystemen. Diese Wissensrepräsentation kapselt Wissen über eine bestimmte Domäne und beschreibt deren Konzepte und Relationen.

OWL: Abkürzung für “Web Ontology Language”. Eine Spezifikation des W3C, um Ontologien anhand einer formalen Beschreibungssprache erstellen, publizieren und verteilen zu können.

Personas: Eine Persona ist eine fiktive Person mit typischen, konkreten Eigenschaften einer Benutzergruppe. Personas veranschaulichen eine Sammlung von Charakteristika und erleichtern die Kommunikation über Benutzerprofile.

Plugin: Ein Plugin ist ein Erweiterungsmodul für ein Software-Produkt. Es ist keine eigenständige Software sondern ergänzt eine bestimmte Software um zusätzliche Funktionalität.

RDF Schema: Abkürzung für “Resource Description Framework Schema”. Eine Empfehlung des W3C, die RDF um die Möglichkeit erweitert komplexere Beziehungen zwischen Ressourcen beschreiben zu können.

RDF: Abkürzung für “Resource Description Framework”. Eine Empfehlung des W3C, um Metadaten in einer formalen Sprache bereitstellen zu können.

RPC: Abkürzung für “Remote Procedure Call”. Eine Technik zur Netzwerkkommunikation, mit der Funktionsaufrufe über ein Netzwerk auf entfernten Rechnern durchgeführt werden können.

Schnittstelle: Schnittstellen von Programmkomponenten sind eine formale Deklaration über die vorhandenen Funktionen und der Art und Weise wie diese angesprochen werden können.

SDK: Abkürzung für “Software Development Kit”. Eine Sammlung von Programmen und Dokumentationen zu einer bestimmten Software. Diese soll Software-Entwicklern das Erstellen von Anwendungen ermöglichen.

Social Navigation: Mit “social navigation” sind Konzepte gemeint, in denen sich Benutzer bei ihrer Navigation am Verhalten und an Hinweisen anderer Nutzer orientieren können.

UML: Abkürzung für “Unified Modeling Language”. Eine standardisierte Sprache für die Modellierung von Software-Systemen.

Unit-Test: Ein Teil des Softwareentwicklungsprozesses zur Verifikation der Korrektheit von Software-Modulen. Dies ermöglicht ein automatisiertes und reproduzierbares Testen der Software.

XML: Abkürzung für “Extensible Markup Language”. Eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien.

A Anhang

A.1 OWL-Beispiel

Beispiel eines OWL-Dokumentes.

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<rdf:RDF
  xmlns="http://www.masterthesis-schmidt.de/Programmierung.owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.masterthesis-schmidt.de/Programmierung.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="programmierung"/>
  <owl:Class rdf:ID="java">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="sprachen"/>
  </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#sprachen">
    <rdfs:subClassOf rdf:resource="#programmierung"/>
  </owl:Class>
  <owl:Class rdf:ID="methoden">
    <rdfs:subClassOf rdf:resource="#programmierung"/>
  </owl:Class>
  <owl:DatatypeProperty rdf:ID="hatInteressenWert">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#programmierung"/>
  </owl:DatatypeProperty>
  <java rdf:ID="Java">
    <hatInteressenWert rdf:datatype="&xsd;int">100</hatInteressenWert>
  </java>
</rdf:RDF>
```

A.2 Benutzerprofil Fragebogen

Die nachfolgenden Seiten zeigen den Fragebogen, der an die befragten Personen versendet wurde. Er diente zur Ermittlung des Benutzerprofils und der grundlegenden Anforderungen an das System.

Benutzerprofil Fragebogen

Im Rahmen einer Masterarbeit wird ein Programm entwickelt, dass den Benutzern einen komfortablen Zugang zu hochschulintern veröffentlichten Mitteilungen ermöglichen soll.

Dieser Fragebogen wurde entwickelt, um mehr über die zukünftigen Benutzer des Systems zu erfahren. Die von Ihnen angegebenen Informationen tragen dazu bei, das Design und die Qualität der Anwendung zu verbessern.

Die Befragung ist **anonym** d.h. **Ihre E-Mail-Adresse wird nicht zusammen mit Ihrer Antwort auf den Fragebogen gespeichert bzw. ausgewertet.** Das Ausfüllen der Fragen sollte nicht mehr als 10 min Zeit in Anspruch nehmen. Bitte Senden Sie den Fragebogen über den "Fragebogen per E-Mail senden"-Knopf bis zum **12.11.2007** zurück.

Vielen Dank für Ihre Teilnahme.

Allgemeine Angaben	
Bitte geben Sie Ihr Alter an.	◯ bis 25 Jahre ◯ 25 - 30 Jahre ◯ 30 - 40 Jahre ◯ über 40 Jahre
Welche Haupttätigkeit üben Sie an der Hochschule aus?	◯ Professor/in ◯ Mitarbeiter/in ◯ Student/in ◯ Andere Tätigkeit
Geben Sie Ihren Fachbereich bzw. Studiengang an.	
Falls Sie ein Student sind geben Sie bitte ihr Semester an.	
Bitte geben Sie an, wie häufig Sie Internetanwendungen nutzen.	◯ Mehrmals pro Tag ◯ Einmal pro Tag ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Selten bis Nie

Anwendungsspezifische Angaben	
Im Gebäude des Departments Informatik und Elektrotechnik gibt es ein "schwarzes Brett" an dem Krankmeldungen, Veranstaltungspläne, GW-Kurse und ähnliches ausgehängt wird. (Es befindet sich im Gang des Erdgeschosses zwischen den Fahrstühlen und der Treppe zur Mensa)	
Krankmeldungen	
Informieren Sie sich über Krankmeldungen?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit sich über Krankmeldungen zu informieren zufrieden? Haben Sie Anmerkungen?	
Hängen Sie Krankmeldungen am Aushang aus?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit Krankmeldungen zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	

Anwendungsspezifische Angaben	
Veranstaltungspläne	
Informieren Sie sich über Veranstaltungspläne?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit sich über Veranstaltungspläne zu informieren zufrieden? Haben Sie Anmerkungen?	
Hängen Sie Veranstaltungspläne am Aushang aus?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit Veranstaltungspläne zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	

Anwendungsspezifische Angaben	
GW-Kurse	
Informieren Sie sich über GW-Kurse?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit sich über GW-Kurse zu informieren zufrieden? Haben Sie Anmerkungen?	
Hängen Sie GW-Kurse am Aushang aus?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit GW-Kurse zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	
Sonstiges	
Informieren Sie sich an diesem "schwarzen Brett" noch über andere Dinge, die nicht oben angeführt sind?	
Veröffentlichen Sie an diesem "schwarzen Brett" noch andere Dinge, die nicht oben angeführt sind?	

Anwendungsspezifische Angaben	
Im Gebäude des Departments Informatik und Elektrotechnik gibt es ein "schwarzes Brett" an dem Gesuche oder Angebote zu Stellen, Wohnungen oder Sonstigem plaziert werden können. (Der Aufsteller befindet sich im Erdgeschoß links neben der Haspa - Filiale)	
Stellengesuche / Stellenangebote	
Informieren Sie sich über Stellengesuche / Stellenangebote?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit sich über Stellengesuche / Stellenangebote zu informieren zufrieden? Haben Sie Anmerkungen?	
Hängen Sie Stellengesuche / Stellenangebote am Aushang aus?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit Stellengesuche / Stellenangebote zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	

Anwendungsspezifische Angaben	
Wohnungsgesuche / Wohnungsangebote	
Informieren Sie sich über Wohnungsgesuche / Wohnungsangebote?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit sich über Wohnungsgesuche / Wohnungsangebote zu informieren zufrieden? Haben Sie Anmerkungen?	
Hängen Sie Wohnungsgesuche / Wohnungsangebote am Aushang aus?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Sind Sie mit dieser Möglichkeit Wohnungsgesuche / Wohnungsangebote zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	
Sonstiges	
Informieren Sie sich an diesem "schwarzen Brett" noch über andere Dinge, die nicht oben angeführt sind?	
Veröffentlichen Sie an diesem "schwarzen Brett" noch andere Dinge, die nicht oben angeführt sind?	

Anwendungsspezifische Angaben	
An der Hochschule gibt es mehrere E-Mail-Verteilerlisten, über die an Informationen gelangt werden kann bzw. über die Informationen an eine bestimmte Personengruppe gesendet werden können. (Bsp.: profs@informatik.haw-hamburg.de, studierende@informatik.haw-hamburg.de, ...)	
E-Mail-Verteilerlisten	
Erhalten Sie Informationen über die E-Mail-Verteilerlisten?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Wenn ja, welche Informationen erhalten Sie?	
Sind Sie mit dieser Möglichkeit Informationen zu erhalten zufrieden? Haben Sie Anmerkungen?	
Versenden Sie Informationen über die E-Mail-Verteilerlisten?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Wenn ja, welche Informationen veröffentlichen Sie?	
Sind Sie mit dieser Möglichkeit Informationen an Personengruppen zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	

Anwendungsspezifische Angaben	
Auf der Hochschulhomepage gibt es einen "News-Bereich" über den Informationen bekannt gegeben werden können. (Bereich auf der rechten Seite: http://www.informatik.haw-hamburg.de/studierende.html)	
News-Bereich	
Informieren Sie sich über Informationen im News-Bereich?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Wenn ja, welche Informationen sich für Sie interessant?	
Sind Sie mit dieser Möglichkeit Informationen zu erhalten zufrieden? Haben Sie Anmerkungen?	
Veröffentlichen Sie Informationen im News-Bereich?	◯ Ja ◯ Nein
Wenn ja, wie häufig?	◯ Täglich ◯ Mehrmals pro Woche ◯ Einmal pro Woche ◯ Mehrmals pro Monat ◯ Einmal pro Monat ◯ Sonstiges bitte angeben:
Wenn ja, welche Informationen veröffentlichen Sie?	
Sind Sie mit dieser Möglichkeit Informationen zu veröffentlichen zufrieden? Haben Sie Anmerkungen?	

Sonstige Angaben	
Bitte geben Sie an, welche zusätzlichen Quellen Sie verwenden um an Hochschulinformationen zu gelangen.	
Wenn Sie dies tun, worüber informieren Sie sich dort?	
Bitte geben Sie an, welche zusätzlichen Quellen Sie verwenden um Hochschulinformationen zu veröffentlichen.	
Wenn Sie dies tun, welche Informationen veröffentlichen Sie dort?	

Vielen Dank für das Beantworten der Fragen.

Fragebogen per E-Mail senden

A.3 Inhalt der DVD-ROM

Dieser Arbeit ist eine DVD-ROM beigelegt.
Auf dieser befinden sich **Ordner** und *Dateien* mit folgendem Inhalt:

- **Masterarbeit:** Dieses Dokument der Arbeit als Datei: *Masterarbeit_Thomas_Schmidt.pdf*.
- **Software:** Die im Rahmen dieser Arbeit entwickelte Anwendung. Eine Installationsanleitung ist in der Datei *Installationshinweise.txt* zu finden.
- **Usability:** Die Vidoaufnahmen der durchgeführten Usability-Tests.

Nach dem Einlegen der DVD öffnet sich ein Web-Browser mit einer Übersichtsseite. Sollte dies nicht geschehen, kann die Seite durch das Öffnen der Datei *inhalt.html* gestartet werden.

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 19. März 2008

Ort, Datum

Unterschrift