Reihe: Telekommunikation @ Mediendienste · Band 11

Herausgegeben von Prof. Dr. Dr. h. c. Norbert Szyperski, Köln, Prof. Dr. Udo Winand, Kassel, Prof. Dr. Dietrich Seibt, Köln, Prof. Dr. Rainer Kuhlen, Konstanz, Dr. Rudolf Pospischil, Brüssel, und Prof. Dr. Claudia Löbbecke, Köln

PD Dr.-Ing. habil. Martin Engelien
Dipl.-Inf. Jens Homann (Hrsg.)

Virtuelle Organisation und Neue Medien 2001

Workshop GeNeMe2001
Gemeinschaften in Neuen Medien

TU Dresden, 27. und 28. September 2001

JOSEF EUL VERLAG
Lohmar · Köln

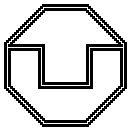
Die Deutsche Bibliothek – CIP-Einheitsaufnahme

Virtuelle Organisation und Neue Medien 2001 / Workshop GeNeMe 2001 – Gemeinschaften in Neuen Medien – TU Dresden, 27. und 28. September 2001. Hrsg.: Martin Engelien ; Jens Homann. – Lohmar ; Köln : Eul, 2001
(Reihe: Telekommunikation und Mediendienste ; Bd. 11)
ISBN 3-89012-891-2

Josef Eul Verlag GmbH
Brandsberg 6
53797 Lohmar
Tel.: 0 22 05 / 90 10 6-6
Fax: 0 22 05 / 90 10 6-88
http://www.eul-verlag.de
info@eul-verlag.de

Printed in Germany
Druck: RSP Köln

Bei der Herstellung unserer Bücher möchten wir die Umwelt schonen. Dieses Buch ist daher auf säurefreiem, 100% chlorfrei gebleichtem, alterungsbeständigem Papier nach DIN 6738 gedruckt.



Technische Universität Dresden
Fakultät Informatik • Institut für Angewandte Informatik
Privat-Dozentur „Angewandte Informatik“

PD Dr.-Ing. habil. Martin Engelien,
Dipl.-Inf. Jens Homann
(Hrsg.)

Dresden, 27./28.09.2001

GeNeMe 2001

Gemeinschaften in Neuen Medien

Workshop zu Organisation, Kooperation und Kommunikation auf der Basis innovativer Technologien

Forum für den Dialog zwischen Wissenschaft und Praxis

an der
Fakultät Informatik der Technischen Universität Dresden

gefördert von der Klaus Tschira Stiftung
gemeinnützige Gesellschaft mit beschränkter Haftung

am 27. und 28. September 2001
in Dresden

http://pdai.inf.tu-dresden.de/geneme
Kontakt: Thomas Müller (tm@pdai.inf.tu-dresden.de)

D.4. Mobiltelefone und Organizer als Zugangsmedien zu Informationssystemen

K. Cords,
L. Gehrken,
K. Panier,
N. Thyssen,
Prof. Dr. J. Raasch
Fachbereich Elektrotechnik/Informatik, FH Hamburg

1. Abstract

Die Teilnahme an Gemeinschaften in Neuen Medien setzt Zugangsgeräte voraus. Die Nutzung leistungsfähiger Universalrechner ist eine verbreitete Möglichkeit, die allerdings eine umfangreiche und kostenintensive technische Ausstattung verlangt. Deswegen ist es notwendig, ergänzend nach Wegen für einen unbeschwerten und kostensparenden Zugang zu suchen.
In diesem Beitrag wird über Software-Engineering-Projekte an der FH Hamburg berichtet. In diesen Projekten wurden Informationssysteme auf einem Server entwickelt und zur Nutzung durch ein Mobiltelefon oder einen Organizer zugänglich gemacht. Hierzu wurden experimentelle und explorative Prototypen entwickelt. Damit entsteht die Möglichkeit, das Informationsangebot von gemeinschaftsbildenden Systemen mit allgegenwärtigen Geräten zu nutzen.
Mit diesen Projekten wird ein Beleg dafür erbracht, dass der Ansatz und die hierzu entwickelte technische Konzeption praktikabel sind. Allerdings wird eine aus Sicht der hier diskutierten Anwendungen erforderliche Leistungssteigerung sowohl der Geräte als auch der benutzten Übertragungsdienste dringend erwartet.

2. Motivation und Vorarbeiten

In vielen Firmen wird zur Zeit der Frage nachgegangen, in welcher Weise Organizer (PDA = personal digital assistent) und WAP-Mobiltelefone (WAP = wireless application protocol) als Zugangsgeräte für operationale, datenbankgestützte Informationssysteme dienen können. Dabei wird langfristig das Ziel verfolgt, Arbeitsschritte der Geschäftsprozessbearbeitung direkt am Entstehungsort der Information ohne besondere technische Ausstattung, wie z.B. Notebook mit Internetanschluss und Drucker, durchzuführen (Beispiel: Außendienstmitarbeiter einer Versicherung beim Kundenbesuch). Die benutzten Geräte werden kaum noch als Computer wahrgenommen. Prozessoren sind

einfach in Gegenstände wie Organizer oder Mobiltelefon mit alltäglicher Semantik als „Leistungsverstärker“ eingebaut. Für diese Sichtweise wurde auch der Begriff des Ubiquitous Computing (s. [27]) geprägt.
An der FH Hamburg können wir dieses Thema im Rahmen von Lehrveranstaltungen, Praktika und Diplomarbeiten bearbeiten. Dabei werden in Lehrveranstaltungen mit Grundlagencharakter Themen aufgegriffen, die auch in der Wirtschaft gegenwärtig intensiv untersucht werden. Informatik wird im Praxiskontext erlernt.
Von den Aufgaben geht damit eine tragfähige Motivation aus: neben den Inhalten des Software-Engineering werden als „added-value“ Kompetenzen in weiteren Gebieten erworben, die für die Positionierung auf dem Arbeitsmarkt unmittelbar interessant sind, z.B. die selbständige Einarbeitung in aktuelle Technologien und marktübliche Systeme.
Im Hinblick auf die Bildung von Gemeinschaften mit Unterstützung durch das Internet sehen wir in der Nutzung der angesprochenen, einfachen Geräte große Chancen: Mobiltelefone und Organizer sind verbreitete Geräte, die neben ihrem sonstigen Wert auch als Zugangsmedien zu Gemeinschaften dienen können. Diese Potentiale sollen untersucht werden.
Für die hier vorgestellten Entwicklungen gibt es Vorarbeiten aus dem SEVERS-Projekt (SEVERS = Software-Engineering für die Versicherungswirtschaft), das über einige Jahre an der FH Hamburg betrieben wurde (s. [20], [4]). Bei früheren Fragestellungen wurde etwa das Tracking untersucht: Kunden sollten ähnlich wie im Logistikbereich den Fortgang von Geschäftsprozessen im Internet verfolgen können. Damals wurde bereits die Nutzung verschiedener Kommunikationsgeräte als weiterer Arbeitsschwerpunkt aufgeführt ([4]S.71).

3. Ein Palm-Organizer als Zugangsgerät für ein Data-Warehouse

Im Sommersemester 2000 wurde von einer Arbeitsgruppe im Rahmen des Praktikums zur Vorlesung Software-Engineering (4. Semester) ein Palm-Organizer (s. [19]) als Zugangsgerät für ein Data-Warehouse auf einem Web-Server genutzt. Da hierfür keine Datenbank-Fortschreibung im Dialog zu realisieren war, konnte auf Transaktionsverarbeitung verzichtet werden. Als möglicher Anwender wurde ein Manager gesehen, der aus einer Sitzung heraus mit den Geräten, die er bei sich trägt (Palm-Organizer und Mobiltelefon), auf das Data-Warehouse seines Unternehmens zugreift.
Insgesamt wurden für dieses Teilprojekt vier Arbeitsgruppen mit den folgenden Aufgaben und Ergebnissen gebildet:

- Realisierung einer Client-Anwendung für den Palm
  Der Client richtet in einem ersten Schritt eine Katalogabfrage nach Art der verfügbaren Daten an den Anwendungsserver, auf dem das Data-Warehouse

lokalisiert ist. Aufgrund der zurückgegebenen Datenauskunft konkretisiert der Benutzer seine Anfrage. Der Server übermittelt in einem zweiten Schritt die abgefragten Daten. Der Client besitzt auch Auswertungsfunktionen (statistische Verteilungsmaße sowie eine grafische Darstellung)

Es wurde eine Abfragesprache als neutrale Schnittstelle realisiert, die prinzipiell eine Anbindung des Palm-Client an ein beliebiges Data-Warehouse ermöglicht, sofern ein spezifischer Parser für die Abfragesprache entwickelt wird. In diesem Projekt wurde die aktuellere Alternative XML (s. [29]) noch nicht benutzt. Der Client wurde mit Smalltalk realisiert.

- Realisierung eines Data-Warehouse für den Client

  In diesem Team stand die Realisierung der Interaktionskomponente zum Client im Vordergrund der Arbeiten. Dazu wurde ein Data-Warehouse mit einem Parser für die Abfragesprache und ansonsten einfacher Funktionalität realisiert. Dieser Prototyp hatte experimentellen (vgl. [12-84]) Charakter, denn es sollte die Nutzbarkeit der Technologie evaluiert werden.

- Realisierung eines komfortableren Data-Warehouse

  Diese Arbeitsgruppe entwickelte parallel zu den anderen ein eigenes Data-Warehouse mit umfassender Auswertungsfunktionalität und Internet-Abfrage auf der Basis des Datenbanksystems Oracle. Dieser Prototyp verfolgte eine explorative (vgl. [7]) Zielsetzung: es sollte untersucht werden, ob die realisierten Anwendungs-konzepte für einen Anwender akzeptabel sind.

- Realisierung eines objektorientierten Data-Warehouse

  Die vierte Gruppe verließ in der Persistenzebene den Bereich der relationalen Datenbanktechnologie. Die im Data-Warehouse wiedergegebenen Informationen beschreiben Informationsträger mit mehreren beschränkten Merkmalsdimensionen. Daher ist der Informationsraum als Quader vorstellbar, hier ist der Begriff des OLAP-Würfel (Online Analytical Processing, s. [16]) verbreitet.

  Bei Nutzung einer relationalen Datenbank wird in der Regel dieser OLAP-Würfel nicht im System realisiert, sondern lediglich direkt in ein Schema umgesetzt, das sich an den Möglichkeiten des Datenbanksystems orientiert. Bei objektorientierter Realisierung wird der Vorteil gesehen, dass der OLAP-Würfel direkt im System modelliert ist. Also gibt es eine Klasse „Würfel“, die einen Container von „Zellen“ hält. Eine Auswertung des Data-Warehouse ist stets zusammensetzbar aus zwei generischen Operationen: „Datenauswahl über Dimensionseinschränkung“ und „Auswertung eines Würfels, (z.B. Mittelwert über alle Würfelzellen)“. Für die Realisierung wurde schließlich Java als Programmiersprache und Objectivity (s. [17]) als objektorientiertes Datenbanksystem benutzt.

Im Ergebnis zeigte dieses Projekt folgende Erkenntnisse:

- Es ist praktikabel, statistische Auswertungen aus dem Data-Warehouse auf dem Palm-Client darzustellen, wenn auch die Anzeigemöglichkeiten begrenzt sind.
- Die Schnittstelle ermöglicht eine Anpassung des Palm-Clients an beliebige Data-Warehouse-Produkte.
- Wir konnten eine leistungsfähige, explorative Alternative zum einfacheren, experimentellen Anbindungsprototyp vorweisen.
- Und wir konnten einen Weg aufzeigen, wie eine objektorientierte Modellierung und Realisierung eines Data-Warehouse durchzuführen ist.

Die Arbeiten wurden von Teilnehmern der Lehrveranstaltung im Rahmen ihres Hauptpraktikums als Mitarbeiter einer Firma fortgesetzt. Daneben hat es eine Diplomarbeit gegeben, in der die Nutzung eines Psion-Client als Frontend für Geschäftsprozesse im Außendienst von Versicherungen näher untersucht wurde [25].
In diesem Beitrag soll jedoch hauptsächlich auf die Weiterführung des Themas, der Einsatz von WAP-Mobiltelefonen, eingegangen werden.

4. Ein WAP-Mobiltelefon als Kommunikations-Endgerät für Anwendungen

Im Wintersemester 2000/2001 wurden die Arbeiten von einer neuen Semestergruppe im Software-Engineering-Praktikum fortgesetzt. Hierüber wird in diesem Beitrag schwerpunktmäßig berichtet.
Durch Internet-Anwendungen entstehen Gemeinschaften. Im Bereich des Studiums treffen drei Gemeinschaften aufeinander: Die Studierenden, die Lehrenden und die Wirtschaft. Im Rahmen des Software-Engineering-Praktikums setzten wir uns zum Ziel, ein Informationssystem zu entwickeln, welches die Zusammenarbeit dieser Gemeinschaften effizient unterstützt.
Ein Anwendungsbereich ist die Darstellung von Informationen über Firmen und deren Angebote für Werkstudenten, Praktikanten, Diplomanden und Absolventen (Festanstellung nach Beendigung des Studiums). Es ist nützlich, wenn derartige Informationen über das Internet für den studentischen Nutzer des Systems einfach per Mobiltelefon abrufbar sind.

4.1 Geschäftsprozesse

Ausgangspunkt der Systementwicklung war eine Erkundung der für die Anwendung relevanten Geschäftsprozesse und die Konkretisierung der gemeinsam gefassten

Aufgabenstellung in einer Systemvision. Die folgenden Anforderungen gaben zur Geschäftsprozessmodellierung Anlass:

- Eine Firma soll sich in einer Firmenliste der Hochschule eintragen können und Angebote für Werkstudenten, Praktikanten, Diplomanden und Absolventen publizieren können.
- Ein Administrator an der Hochschule soll diese Angaben verwalten und weitere Informationen wie zum Beispiel persönliche Kontakte und Mitgliedschaften im Förderverein pflegen können.
- Jeder Studierende der Hochschule soll Firmeninformationen und Angebote komfortabel nach individuellen Wünschen und Neigungen abfragen können.

4.2 Systemvision

Die geschilderten Anforderungen lassen sich gut mit Hilfe einer Internetanwendung erfüllen. Durch die direkte und schnelle Kommunikation entsteht die Möglichkeit, eine aktuelle und zugleich umfangreiche Datenbasis verfügbar zu machen. Durch Verwendung von Standardhilfsmitteln (Browser) entsteht eine leichte Zugänglichkeit und Handhabbarkeit.

Es wurden mehrere Möglichkeiten erkannt, auf das System zuzugreifen. Für unsere Anwendung boten sich an

- Webinterface auf Basis von interaktiven, dynamischen HTML-Seiten
- WAP-Mobiltelefon
- Eigenständige Dialoganwendung für lokale Nutzung ohne Internet

Entsprechend dieser Nutzungsarten entstanden folgende Präferenzen:

- Registrierung der Firmen und Publikation ihrer Angebote: Webinterface
- Information für Studierende: Webinterface, WAP-Mobiltelefon
- Administration: Eigenständige Dialoganwendung

5. Systemarchitektur und Konzeption

Auf Basis der Anwenderanforderungen entsteht die Architektur unter Beachtung aktueller Entwurfsregeln des Software-Engineering:

- Die Interaktionskomponente muss von der Funktionskomponente getrennt werden. Hier ist schwache Kopplung erforderlich; es darf nur Referenzen von der Interaktion zur Funktion geben; es müssen Rückkopplungsmechanismen implementiert werden, aufgrund derer die Interaktion von Änderungen innerhalb der Funktion erfolgt (MVC-Konzept, s. [15] bzw. Observer-Pattern, s. [8], vgl. auch [30]).

- Die Komponenten zur Benutzerkommunikation (eigenständige Dialoganwendung, Webinterface, WAP-Interface) sollen in die Interaktionskomponente eingebettet werden, aber unabhängig voneinander und von der Funktionskomponente entwickelt werden.
- Die Entkopplung soll möglichst neben der schwachen Kopplung eine Verteilung mit Orts- und möglichst auch Sprachtransparenz der beteiligten Komponenten erzeugen. Daher benutzt die Entkopplungsschicht XML und TCP/IP (Entwurfsentscheidung).
- Die Funktionskomponente muss eine interne Substruktur aufweisen, die sowohl die zu benutzende Datenbank nach außen verbirgt als auch auf Ebene der Anwendung objektorientiertes Arbeiten ermöglicht. Dadurch wird eine Anpassungsschicht (*DBInt*) für die Datenbank an die objektorientierte Anwendung erforderlich.

Diese Grundsätze führen zwingend zur Architektur, wie sie in Abb.1 wiedergegeben ist. Ein Hauptziel des Projektes war, neben professionellem Einsatz von Methoden des Software-Engineering einige aktuelle Technologien kennenzulernen. Insofern bildet das System einen Musterfall für wesentlich komplexere Aufgaben. Das vordergründige Anwendungsziel, Studierende über Firmen zu informieren, hätte man auch einfacher erreichen können.

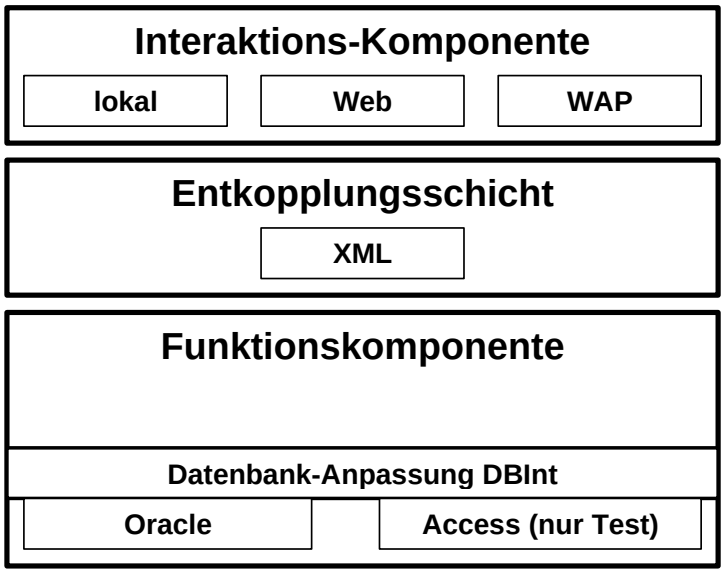


Abbildung 1: Architektur der Anwendung

5.1 Die Entkopplungsschicht

5.1.1 XML

Für die Realisierung der Entkopplungsschicht liegt die Verwendung von XML (eXtensible Markup Language, s. [29], [2]) nahe. Dafür werden folgende Gründe gesehen:

- Universelle Beschreibungssprache
- Unkomplizierter Aufbau der Sprache
- Aktuelle Technologie
- Nähe zur Objektorientierung (ein XML-Dokument beschreibt eine Objektstruktur)

Es bleiben noch die folgenden Fragen und Probleme zu klären:

- Strukturen in XML
  Wie übersetzt man die zu transportierenden Objekte und Methoden in ein XML-Dokument? Ist eine Eigenentwicklung erforderlich oder können vorhandene Komponenten benutzt werden?
- Generischer Informationsaustausch
  Soll der Informationsaustausch zwischen den Komponenten spezialisiert für das semantische Modell entwickelt werden oder generisch und damit anwendungsunabhängig?
- Schnittstellen beibehalten
  Die Benutzungsschnittstelle und der Funktionskern können entweder in einer Programmeinheit zusammengebunden sein oder räumlich getrennt zum Beispiel als Client/Server-Anwendung ausgelegt sein. Die Benutzungsschnittstelle und der Funktionskern sollen von derartigen Lokalisierungsfragen unabhängig sein.
- Sicherheit und Übertragungsprotokoll
  Bei einer Client/Server-Anwendung stellt sich generell das Problem der Sicherheit und des Übertragungsprotokolls.

5.1.2 Entwurfsentscheidungen der Entkopplungsschicht

Diese Fragen und Probleme konnten unter Etablierung der folgenden Designentscheidungen gelöst werden:

- **Java**
  Die Realisierung des gesamten Projektes erfolgte in Java. Dadurch entstanden keine besonderen Integrationsprobleme.
- **JDOM**
  Um XML-Dokumente berarbeiten zu können benötigt man einen Parser. XML-

Parser lassen sich nach zwei Kriterien unterscheiden, abhängig davon, welche Schnittstelle sie zum Zugriff auf das XML-Dokument anbieten (SAX oder DOM).

- *Simple API for XML (SAX)* [21] ist schnell und einfach, hat allerdings den Nachteil, dass kein Objektmodell zur Verfügung steht, auf das man nach dem Parsen zugreifen kann. SAX ist ideal für alle Anwendungen, die das XML - Dokument nur einmal „durchlesen“ sollen, beispielsweise um es anzuzeigen.
- *Document Object Model (DOM)* [3] hingegen ist bedeutend langsamer, bietet aber die Möglichkeit der einfachen Verarbeitung eines XML-Dokuments, da zum Dokument ein Objektmodell erzeugt und im Speicher gehalten wird. DOM ist also bei Anwendungen zu verwenden, die interaktiv auf das XML–Dokument zugreifen müssen (Editoren) oder die das Dokument mehrfach brauchen.

Ein geeignetes Werkzeug für XML und Java ist JDOM (OpenSource-API, s.[11], [10]). JDOM versucht die Vorteile von DOM und SAX zu kombinieren. Die teilweise recht umständlichen Methoden zur Generierung und Manipulation eines DOM werden dabei vereinfacht. Anders als bei SAX unterstützt JDOM auch einen direkten Zugriff auf das XML – Dokument, wobei es keine Notwendigkeit gibt (wie bei DOM), das ganze Dokument im Speicher zu halten. Darüber hinaus ist JDOM, im Unterschied zu DOM und SAX, speziell auf die Sprache Java zugeschnitten und somit in einigen Bereichen stark optimiert.

- **TCP/IP**
  Als Übertragungsprotokoll wurde TCP/IP benutzt. Dies ist eine entscheidende Voraussetzung für weltweite Verteilung im Internet und Plattformunabhängigkeit.
- **SSL**
  Sicherheit wurde, vorbehaltlich vertiefter Untersuchung, mittels Verschlüsselung durch SSL (s. [22]) gewährleistet.

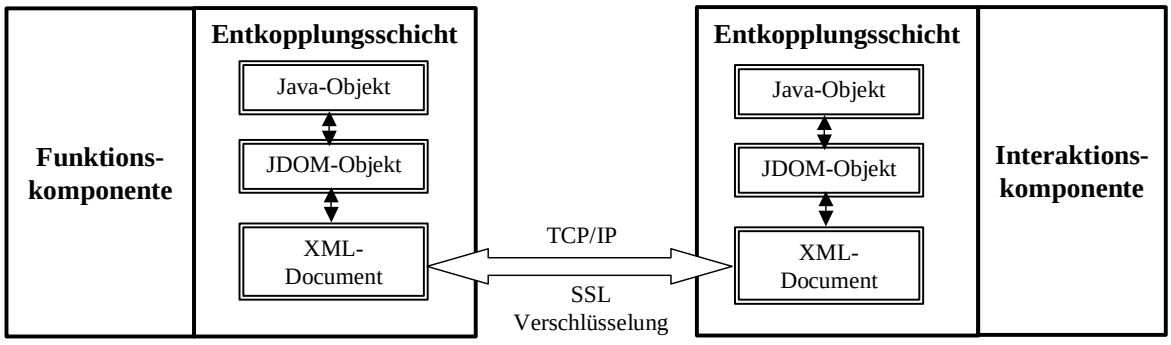


Abbildung 2: Objekttransformation in der Entkopplungsschicht

5.1.3 Funktionsweise der Entkopplungsschicht

Um eine Unabhängigkeit der Entkopplungsschicht von der Funktionskomponente bezüglich Realisierung und Weiterentwicklung zu erreichen, müssen Informationen transparent übertragen werden. Daher dürfen Klassen- oder Methodennamen in der Entkopplungsschicht nicht bekannt sein. Hier ist also nur eine *generische* Lösung praktikabel, die direkte Lösung ist zu verwerfen.
Die generische Lösung ist in der Lage, völlig unbekannte Objekte nach XML umzuwandeln und das XML-Abbild des Objektes wieder zurück in das Objekt zu wandeln (vgl. Abb. 3).

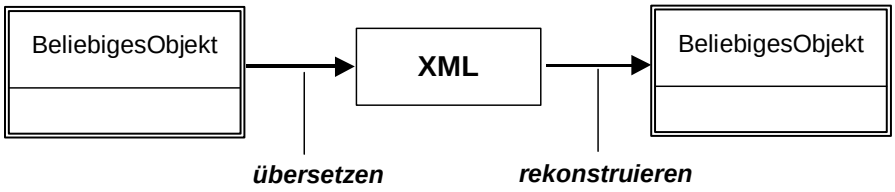


Abbildung 3: Objekttransfer durch generischen Informationsaustausch

Mit Hilfe der Java Reflection Classes können Attribute unbekannter Objekte rekursiv bis zur Ebene der elementaren Datentypen erfasst und in XML umgesetzt werden, wobei die Klassennamen als XML-Tags wirken. Auf Empfängerseite werden durch Aufruf der "invoke"-Methode bei bekannter Klasse (XML-Tags) die Objekte instantiiert und deren Attribute gesetzt.
Für dieses Vorgehen sind gewisse Einschränkungen notwendig. Zum Beispiel müssen die Klassen der umzuwandelnden Objekte sich an bestimmte Namenskonventionen halten und zyklenfrei sein .

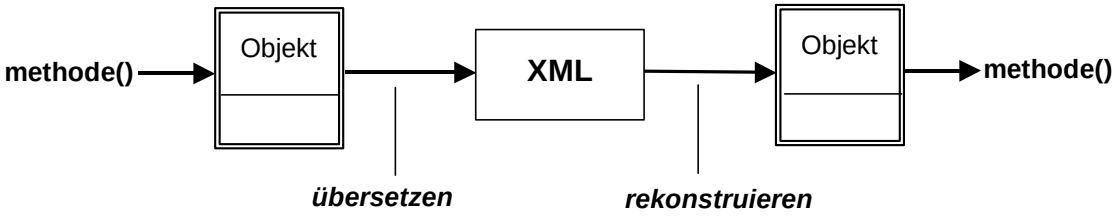


Abbildung 4: Übermittlung eines Methodenaufrufs

Das entwickelte Framework erlaubt auch, Methoden über Rechnergrenzen hinweg aufzurufen. Hierzu wird aus dem Methodenaufruf ein Objekt erzeugt und entsprechend dem obigen Verfahren nach XML übersetzt. Beim Empfänger wird das Objekt rekonstruiert und die Methode evaluiert (s. Abb. 4).

5.2 Die Funktionskomponente

Die Funktionskomponente hat die Aufgabe, das semantische Modell des Anwendungsbereiches zu kapseln und alle für die Werkzeugkonstruktion benötigten Dienste anzubieten (vgl. [30]). Im einzelnen sind folgende Aufgaben zu realisieren:

- Domainmodell und die von der Interaktionskomponente benötigten Dienste
- Datenbank (Persistenz)
- Dynamisches Modell des Anwendungsbereiches

5.2.1 Domainmodell

Um die geforderten Verhältnisse abzubilden, wurde das in Abb. 5 wiedergegebene Klassenmodell entwickelt.

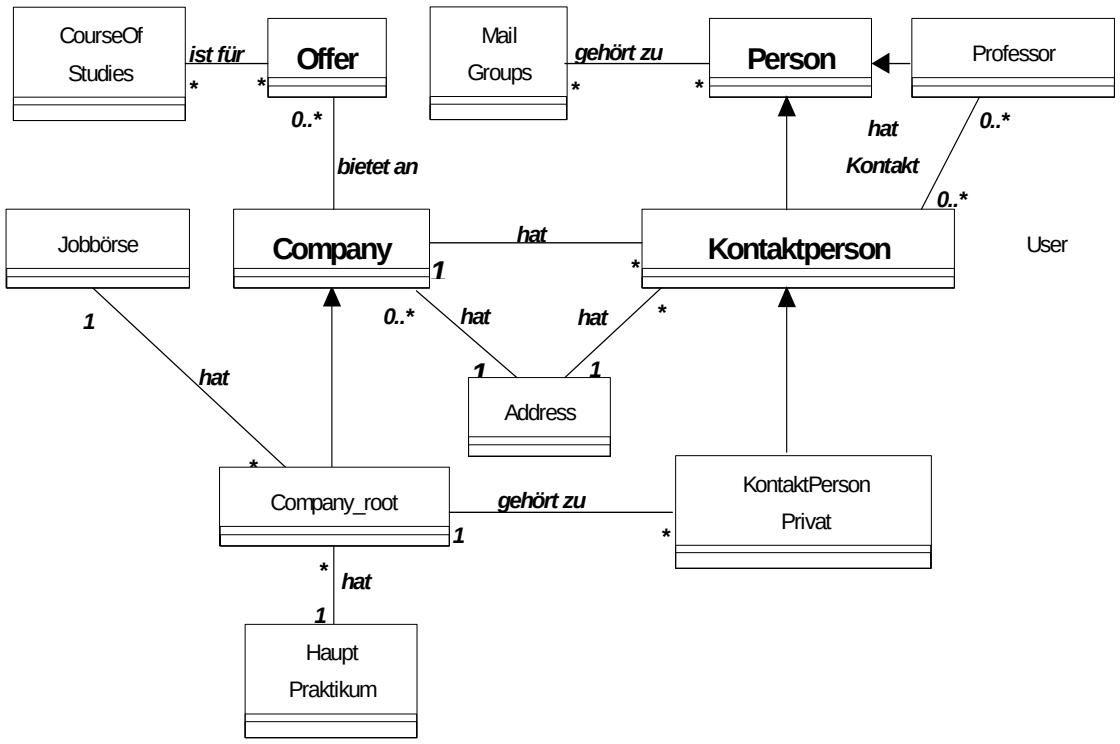


Abbildung 5: Klassenmodell

In dem Diagramm (Abb. 5) wurden die für das Ziel besonders wichtigen Klassen gesondert herausgestellt. Der Kernpunkt unserer Applikation sind die Angebote von Firmen für Studierende. Dementsprechend stehen für uns die Firmen (Klasse „Company“) mit ihren Angeboten (Klasse „Offer“) und Ansprechpartnern (Klasse „Kontaktperson“ bzw. „Person“) im Mittelpunkt, da es diese Informationen sind, welche die Studierenden interessieren.

Weitere Klassen modellierten wir, um zusätzliche Funktionen zu integrieren. So entstanden Klassen wie Jobbörse und Hauptpraktikum, welche es dem Administrator erlauben, eine Firma präziser zu beschreiben. Für weitere Features des Systems wurden Klassen wie MailGroups sinnvoll.

5.2.2 Datenbank

Die Persistenz der Anwendungsobjekte wurde weitgehend von der Anwendungsfunktionalität entkoppelt (s. Abb 6). Die Klasse „ObjectLayer“ stellt die Objekte im Resultset für die Bearbeitung durch die Klasse „FK“ bereit. Von dem benutzen Datenbanksystem wird zusätzlich durch eine Klasse DBInt abstrahiert. Die Klassen „ObjectLayer“ und „DBInt“ bilden damit zusammen eine Materialverwaltung (vgl. [30]S.300ff).

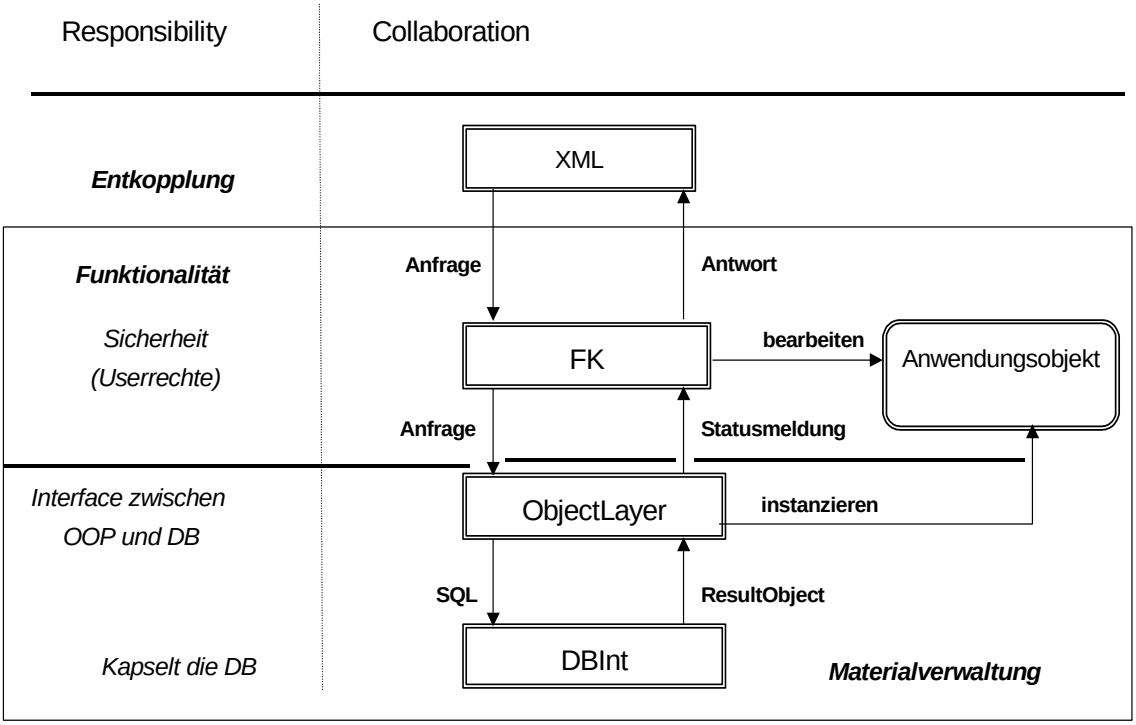


Abbildung 6: Kooperationsdiagramm eines Datenzugriffes

Die Kapselung durch die Klasse DBInt ermöglichte den problemlosen Austausch des Datenbanksystems. Sogar die Verwendung von Datenbank-spezifischen Features wie z.B. Stored Procedures ist möglich, da diese komplett von der Klasse DBInt verwaltet und gekapselt werden. Dass dieses Konzept aufgeht, zeigte sich schon in der parallelen Verwendung von Oracle Version 8 (s. [18]) auch Access 97 als Datenbanksystem.

Dabei diente Access lediglich als Testumgebung für die Modulentwicklung. Die Integration an der Hochschule wurde mit Oracle durchgeführt.
Für den Zugriff kamen unterschiedliche Verfahren zum Tragen. So wurde die Oracle Datenbank über TCP/IP angesprochen, während bei Access die ODBC Schnittstelle zum Einsatz kam.

5.2.3 Dynamisches Modell

Abbildung 6 zeigt als Kooperationsdiagramm den Ablauf von Datenbereitstellungen aus der Datenbank. Die Verbindung zwischen der relationalen Datenbank und der objektorientierten Anwendung wird durch die Klasse „ObjectLayer“ hergestellt. Der „ObjectLayer“ wandelt eine Anfrage nach Daten in eine geeignete SQL-Anweisung um, die an die Klasse „DBInt“ weitergereicht wird. Aus dem resultierenden Ergebnis rekonstruiert der „ObjectLayer“ dann wieder Objekte, welche für die Klasse „FK“ bereitgestellt werden.
Die Persistenz von Objekten wird mittels eines Java-Interfaces realisiert, welches jede Klasse des Domainmodells implementieren muss. Dieses Interface definiert die erforderlichen Kriterien, die ein Objekt für eine erfolgreiche Transformation in eine Menge von Datensätzen zu gewährleisten hat.

5.3 Die Benutzungsschnittstellen

5.3.1 Die Benutzungsschnittstelle zur Administration

Anforderungen

Für die Administration sollte eine Anwendung entstehen, welche die vollständige Nutzung und Verwaltung des Systems ermöglicht.
Die Anforderungen waren:

- Implementierung aller zuvor genannten Userfunktionalitäten, also das Erfassen und Suchen von Firmen und deren Angebote
- Erfassung von Zusatzkriterien, die nur dem Administrator zugänglich sind, z.B. bestimmte Statusflags oder Kommentare
- Verwertung der Suchergebnisse als Datenquellen für Serienbriefe und Emails
- Reporterstellen für Angebote von Praktika oder Diplomarbeiten
- Administration des gesamten Systems
    - Änderung und Erfassung von Stammdaten wie neue Professoren und Administratoren

- Konfiguration der Web- und WAP-Server sowie der Kommunikationskomponenten

Technologie

Da in allen Teilprojekten Java benutzt wurde, bot sich Swing als Basis für die GUI-Komponenten an. Die Email und Serienbrieffunktionalitäten sollten durch eine universelle Anbindung an existierende Anwendungen wie Word, StarWriter oder Netscape realisiert werden, da die Entwicklung eigener Email-Clients oder Textverarbeitungssysteme zu aufwändig gewesen wäre.
Um eine einheitliche grafische Benutzungsoberfläche bei paralleler Programmierung zu entwickeln, legten wir Richtlinien fest, zum Beispiel für die Anordnung von Elementen in einem Fenster oder den Aufbau einer Titelzeile. Um eine gute Wartbarkeit der Applikation zu erreichen, wurde das Zusammenwirken der Klassen mit Hilfe der CRC-Vorgehensweise (s. [1]) geplant. Der Zugriff auf die persistenten Daten erfolgt nur zentral im Hauptfenster. Alle anderen Fenster starten selbst keine Anfragen, sondern erhalten ihre Daten vom aufrufenden Fenster. So kann zu jeder Zeit sichergestellt werden, dass die Daten konsistent sind.

5.3.2 Die Web Benutzungsschnittstelle

Anforderungen

Jeder Nutzer (z.B. Student, Firma, Administrator) erhält über einen beliebigen Browser Zugang zu den als Partner eingetragenen Firmen sowie deren Angebote. Durch diese Offenheit des Systems entsteht eine Interessengemeinschaft aller beteiligten Personen. Mit geringem Aufwand können Angebote wie Jobs, Praktikantenplätze und Diplomarbeiten einer großen Anzahl von Studierenden unterbreitet werden. Es wird offensichtlich, welche Firmen Partner der Hochschule sind und sich aktiv an der Ausbildung der nächsten Generation von Informatikern und Informatikerinnen beteiligen.
Die Firmen werden in einer Liste dargestellt, aus der man eine Firma wählen kann, um dann ihre Detailinformationen zu erhalten.
Um in der Firmenpartnerdatenbank aufgenommen zu werden, muss zunächst zum Schutz der Daten vor unerlaubten Änderungen eine Registrierung vorgenommen werden. Neben verschiedenen Angaben zur Firma ist ein Login-Name sowie ein Passwort festzusetzen. Danach ist es für die Firma jederzeit möglich, nach dem Login die eigenen Daten und Angebote zu verändern.
Zum Nutzen der Studierenden ist es möglich, alle Angebote fimenunabhängig in einer Liste einzusehen. Die Auswahl eines Angebotes führt zu Detailinformationen der Firma mit Angabe der Kontaktperson.

Technologie

Für die Realisierung eines dynamischen Webauftrittes gibt es mehrere Alternativen, zum Beispiel:

- **Common Gateway Interface** (CGI, s. [14])
  Das CGI spezifiziert den Datentransport vom Web-Server zum Serverprogramm. Für jede HTTP-Anfrage startet der Web-Server eine neue Instanz des Serverprogramms, so dass bei mehreren Anfragen mehrere Prozesse ein- und desselben Programms existieren können. Das hat den Vorteil, dass der Web-Server durch den Absturz einer Anwendung nicht selbst zum Absturz gebracht werden kann und den Nachteil, dass die Rechenzeit durch den Prozesswechsel erhöht wird. Das CGI ist seit 1995 im Einsatz.
- **Java Servlets** (s. [13])
  Java Servlets bieten ein hohes Maß an Flexibilität, da sie wegen der Programmiersprache Java nicht von der Plattform des Web-Servers abhängig sind.
  Zusätzlich bietet das Servlet API Unterstützung für Session-Managment und Cookies an. Bei jeder neuen Anfrage vom Browser wird ein neuer Thread, jedoch kein neuer Prozess erzeugt, wodurch die Performance verbessert wird.
- **Proprietäre Server-API** (s. [14])
  Bei den Server-API (Anwendungsprogrammierschnittstellen direkt von den Herstellern zu ihren eigenen Web-Servern) werden die Programme (z.B. Netscape Server Application Programming Interface von Netscape, Internet Server Application Programming Interface von Microsoft) direkt in den Web-Servern eingebunden. Dies hat den Vorteil der höheren Geschwindigkeit, kann jedoch auch zum Absturz des Servers durch die Programme führen. Nachteilig ist auch die Bindung an einen speziellen Web-Server.

Entscheidung für Java-Servlets

Es wurden Java Servlets eingesetzt, weil sie eine hohe Performance bieten und durch die Plattformunabhängigkeit portabel sind. Außerdem konnte so eine neue Technologie kennengelernt werden. Die Integration in das Gesamtsystem wurde vereinfacht, da alle Gruppen Java als Programmiersprache verwendeten.

Neben dem Java Servlet Development Kit JSDK [12] wird ein Web-Server sowie eine Servlet – Engine benötigt. Aus der Vielzahl der angebotenen Engines haben wir uns für die Engine Jakarta Tomcat [23] entschieden, diese ist weit verbreitet, kostenfrei erhältlich (Open Source) und wird von Sun direkt unterstützt. Damit kamen wir zu der in Abb. 7 wiedergegebenen Architektur.

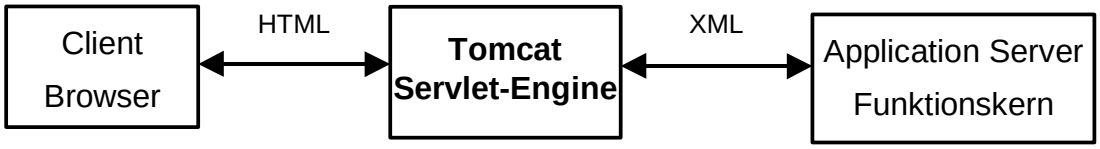


Abbildung 7: Architektur der Kommunikationsverbindung

In dieser Konstruktion übernimmt die Tomcat Servlet-Engine die Funktion eines Web-Servers.

5.3.3 Die WAP-Benutzungsschnittstelle

Anforderungen

Jeder Nutzer des Systems sollte die Möglichkeit haben, auf Firmen und Kontaktpersonen zugreifen zu können. Der Idealfall ist es, diese Informationen direkt vom Mobiltelefon aus abzurufen, als wären diese Bestandteil des im Telefon integrierten Telefonbuchs. Dazu gehört auch die Funktion des sofortigen Anrufes bei dem gewünschten Gesprächspartner.
Um dieses Verhalten zu realisieren, muss von der Applikation ein Frontend zur Verfügung gestellt werden, welches über Suchmasken für Firmen und Personen verfügt. Die Suchergebnisse sollten als überschaubare Liste erscheinen, wobei man für jeden Eintrag eine Detailansicht mit den Daten des Kontaktes abrufen kann. Von diesem Punkt aus sollte es möglich sein, eine Verbindung zu dem gewünschten Gesprächspartner direkt herzustellen.

Technologie

Für ein Mobiltelefon bieten sich prinzipiell verschiedene Wege für den Zugriff auf Informationssysteme:

- *Voicebox* [26]
  Bei diesem Verfahren wird ein System angerufen, welches über ein Sprachinterface (z.B.: Sprachfähige Modems, ISDN-Karten oder spezielle Sprachkarten) verfügt. Dieses System kann mittels Spracherkennung oder Betätigen der Tasten des Telefons verschiedene Funktionen ausführen und Texte vorlesen.
- Wireless Application Protocol WAP [WAP]
  Dieses Protokoll definiert einen Zugriff auf Internet Server und ist dem HTTP verwandt. Daher bieten sich für einen Zugriff mittels WAP auf ein Informationssystem die gleichen Technologien wie bei einem Web-Server. Auch

hier gibt es ein spezielles Übertragungsformat der Dokumente, das WML (Wireless Markup Language, [28]), das anstelle von HTML zu benutzen ist. Daraus ergeben sich für den WAP-Server die gleichen technischen Realisierungsmöglichkeiten wie für den Web-Server.

Daher lag es nahe, WAP als Zugriffsmedium zu verwenden und auf Serverseite auch Java-Servlets einzusetzen.

WML

WML ist stark an HTML angelehnt, nur sind dort die Regeln härter gefasst. Wenn ein HTML-Dokument syntaktisch nicht einwandfrei ist, versucht der Browser, das Dokument zu interpretieren, um es dennoch wie gewünscht darzustellen (selbsttätige Korrektur unvollständigen HTML-Codes). Für solch eine Funktionalität ist ein WAP-fähiges Mobiltelefon nicht leistungsstark genug. Deshalb wird bei WML stärker auf die Einhaltung der Syntaxregeln geachtet.

Zusätzlich gibt es noch einige technische Besonderheiten. So nimmt ein WML Dokument den Weg vom Server über den Netzprovider (s. Abb. 8). Hier werden Kommentare entfernt, um die Übertragungsmenge zu reduzieren. Außerdem wird das WML Dokument als Bytecode übertragen. Das Mobiltelefon muss dadurch weniger Daten empfangen und es kann diese mit geringerer Rechenleistung verarbeiten.

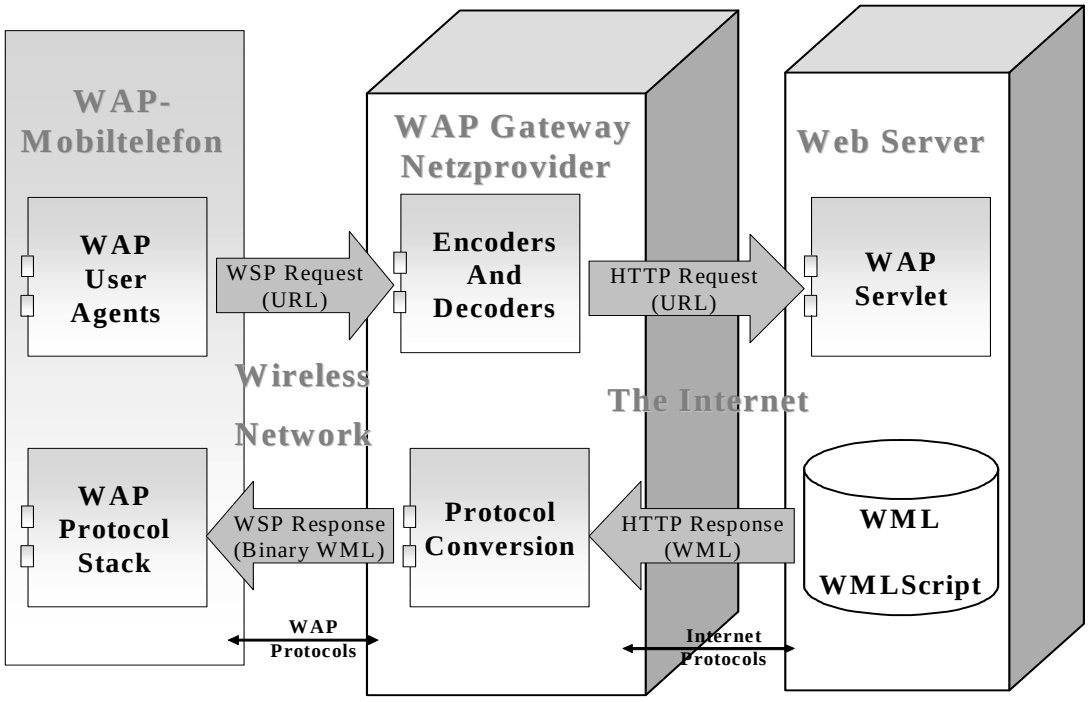


Abbildung 8: Ablauf einer Anfrage durch ein WAP-Mobiltelefon

6. Der Prototyp

6.1 Ziele des Prototyps

In der Hauptsache sollte experimentell erkundet werden, wie ein Mobiltelefon als Zugangsgerät für die Geschäftsprozessbearbeitung genutzt werden kann. Die dabei auftretenden technischen Fragen waren zu klären.
Ein weiteres Ziel bestand darin, zu zeigen, in welcher Weise XML benutzt werden kann, um eine größtmögliche Unabhängigkeit der Komponenten zu erzielen.
Schließlich sollte anhand des Prototyps explorativ diskutiert werden können, ob und wie ein solches System den Anwendungsbereich benutzungsfreundlich unterstützt. Mögliche Anwender sollten in der Lage sein, anhand des Prototyps Entscheidungen zur Durchführung eines Projektes zur Implementierung des Systems zu treffen.

6.2 Benutzungsschnittstellen des Prototyps

6.2.1 Die Administrations-Benutzungsschnittstelle

Nach dem Login gelangt man in das Hauptfenster, welches eine Liste aller Firmen anzeigt.
Die Funktionalitäten der Anwendung werden ausnahmslos vom Hauptfenster aufgerufen. So können Stammdaten bearbeitet oder Emails und Serienbriefe erstellt werden. Die Ergebnisse der Suchfunktionen lassen sich als Reporte ausdrucken oder, wenn es sich um Kontaktpersonen handelt, in Emails oder Serienbriefen weiterverwenden.

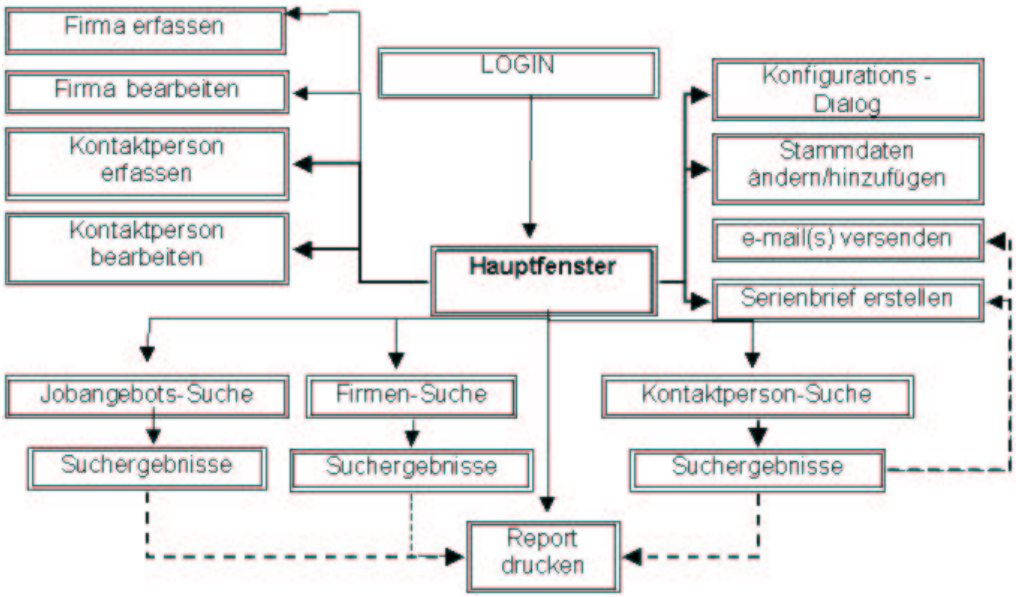


Abbildung 9: Dialogstruktur Administration

Die folgenden Abbildungen zeigen zwei Fenster der Administrationsanwendung. Abb.10 zeigt das Hauptfenster der Anwendung und Abb. 11 die Detailansicht einer Firma zur Bearbeitung der Daten.

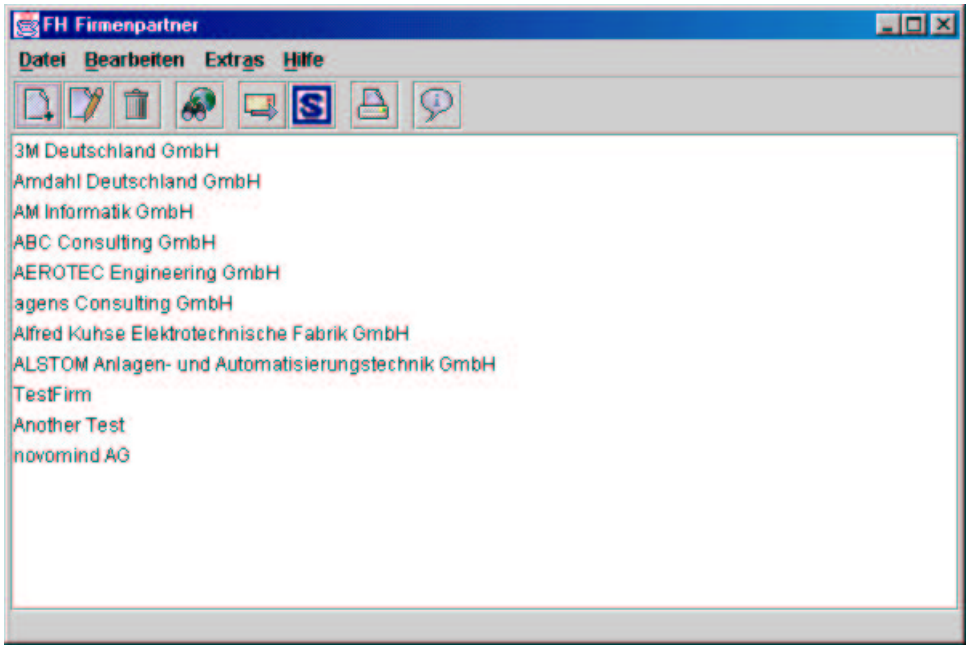


Abbildung 10: Das Hauptfenster der Administrationsanwendung.

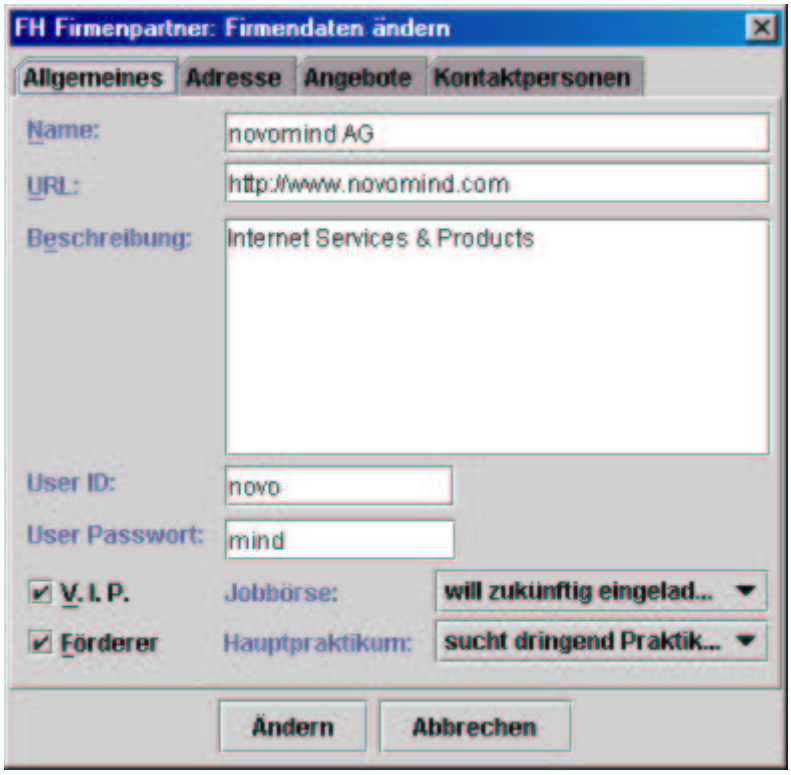


Abbildung 11: Der Firmendialog

6.2.2 Die Web-/WAP-Benutzungsschnittstelle

Folgende Abbildungen illustrieren die Benutzerführung der Web-Seiten.
Eine Firma kann und muss sich vor Eingabe von Angeboten zunächst registrieren, um ein Login zu erhalten. Damit ist sowohl die Accountpflege als auch die Verwaltung von Angeboten möglich (s. Abb.12).

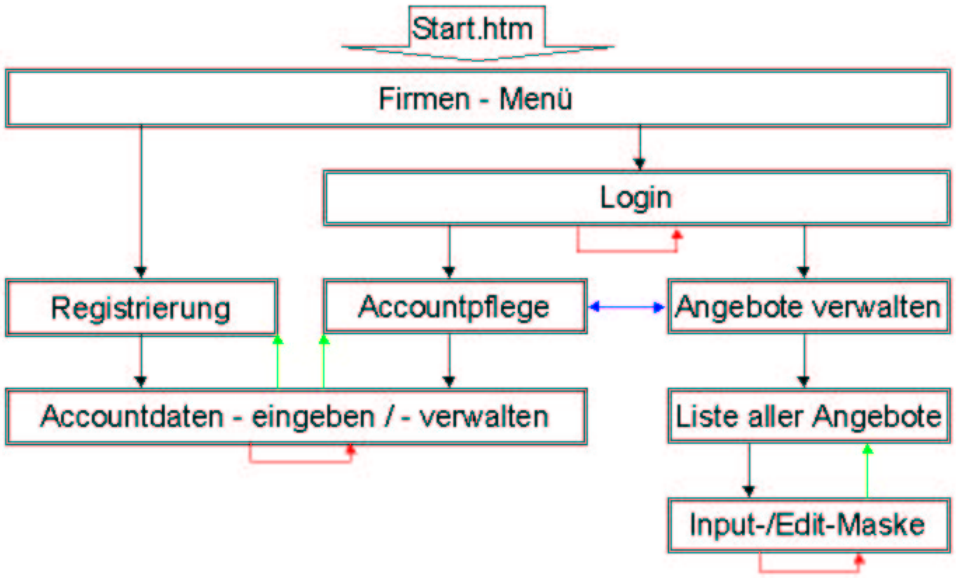


Abbildung 12: Ablauf zur Verwaltung der firmeneigenen Daten

Registrierung einer Firma

Neben den Stammdaten zur Firma wird ein Benutzerkonto definiert. Pflichteingaben sind durch das Zeichen * gekennzeichnet. Nach der Registrierung können die Daten der Firma bearbeitet und Angebote erstellt oder geändert werden. Vorgeschaltet ist dann immer der Login-Dialog.

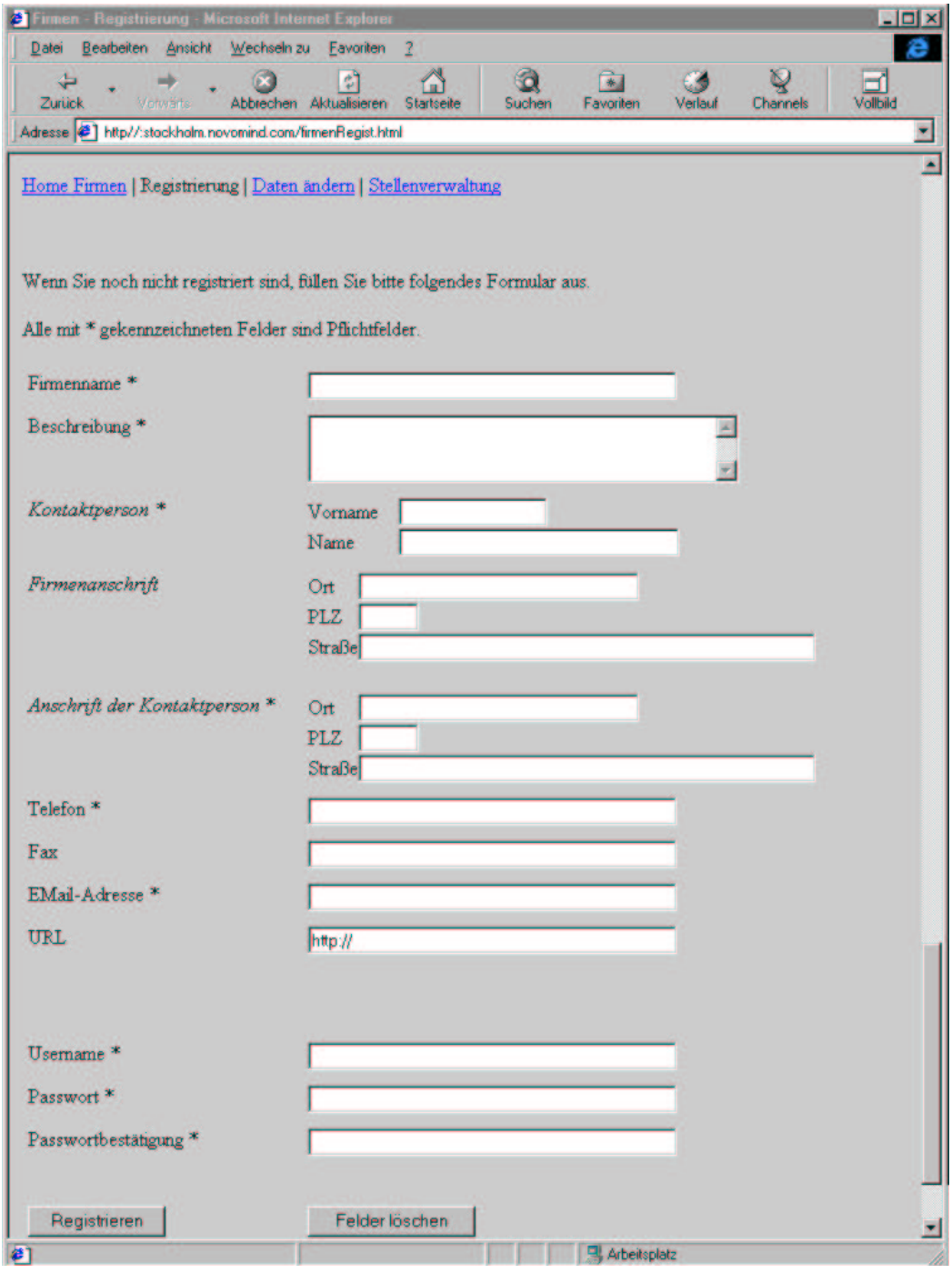


Abbildung 13: Formular zur Registrierung

Verwaltung der Angebote

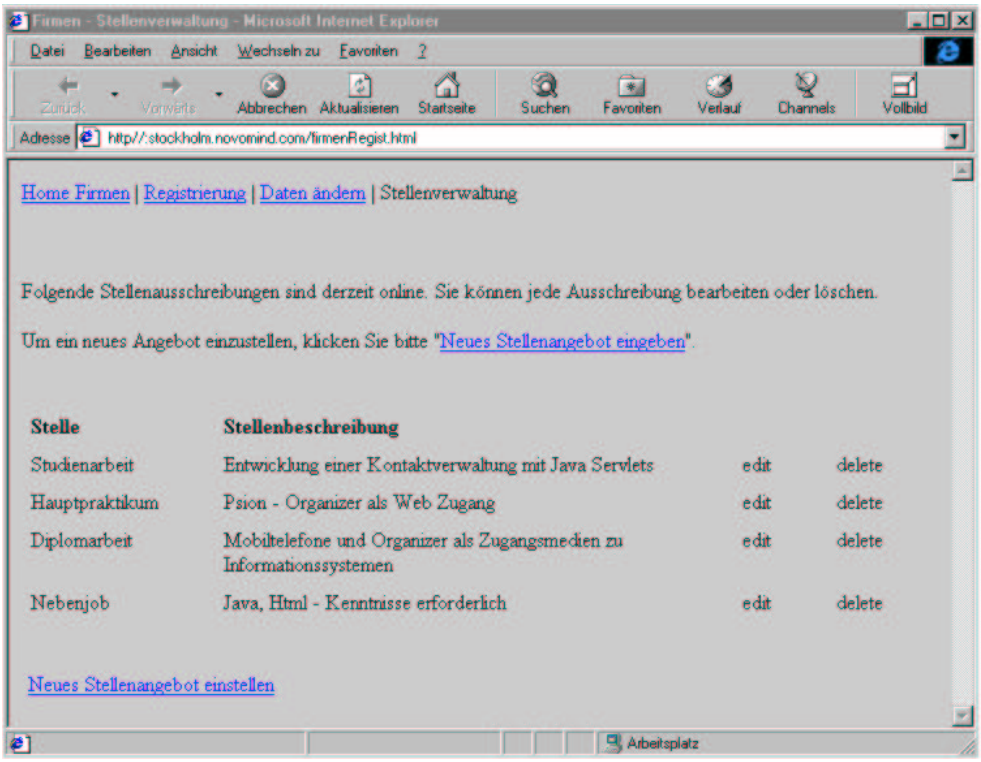


Abbildung 14: Listing der Stellenangebote

Zur Verwaltung der Stellenangebote einer einzelnen Firma werden diese wie abgebildet aufgelistet. Alle Angebote können bearbeitet oder gelöscht werden. Zur Erfassung neuer Angebote muss der entsprechende Link angeklickt werden.

Suchverfahren für Studierende

Studierenden werden verschiedene Suchverfahren angeboten (s. Abb.15):

- Gezielte Suche einer Firma
- Anzeige aller Firmen mit Zugriffsmöglichkeiten auf Details
- Anzeige von Jobangeboten

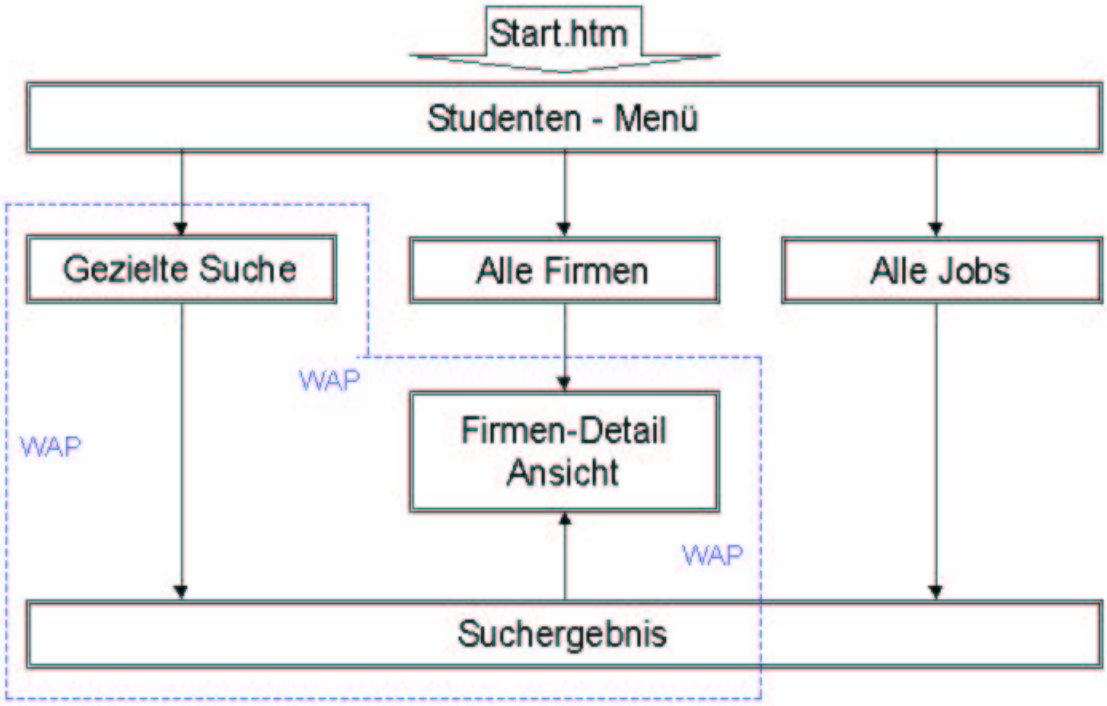
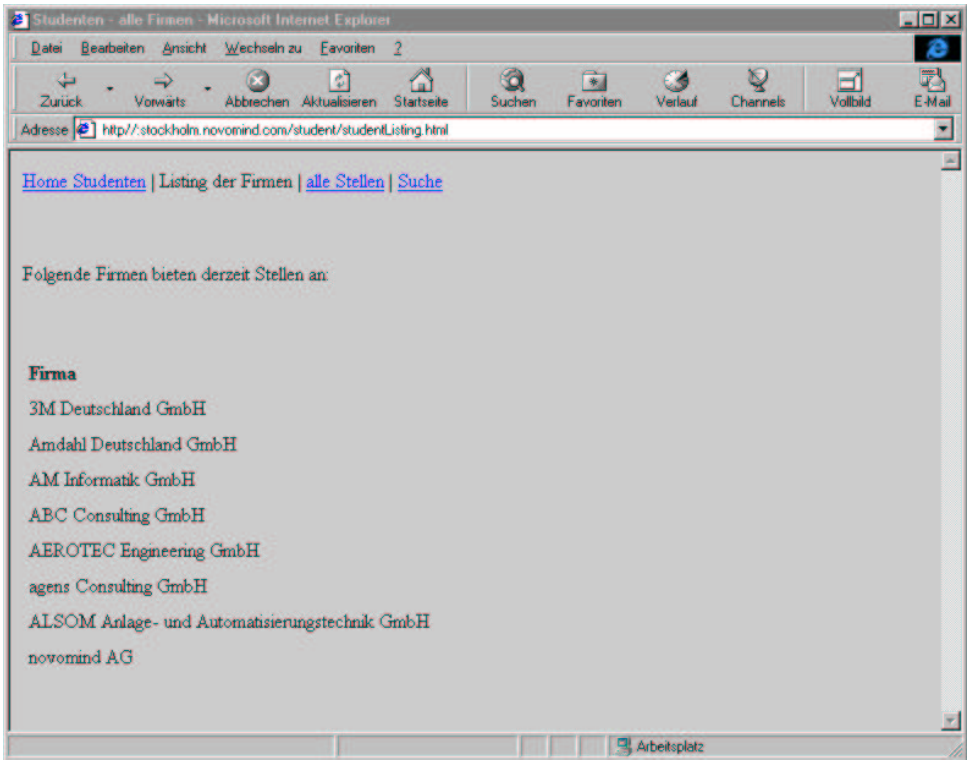


Abbildung 15: Ablauf der Seite zur Suche von Angeboten unter Einschluß der WAP Funktionalität



Von hier aus ist eine Detailansicht zur Firma erreichbar.

6.2.3 Emulation der WAP-Benutzungsschnittstelle

Im folgenden wird ein Beispieldialog über ein WAP-Mobiltelefon in einer Emulation (gelon.net [9]) wiedergegeben.

Über das Mobiltelefon ist die Suche nach einer Firma oder nach einer Person möglich. wird im ersten Schritt eine Firma gesucht und im zweiten Schritt angezeigt. In beiden Fällen ist ein Suchkriterium einzugeben. Die Treffer werden in einer Ergebnisliste angezeigt. Eine Selektion in der Ergebnisliste führt dann zu den Detailangaben.

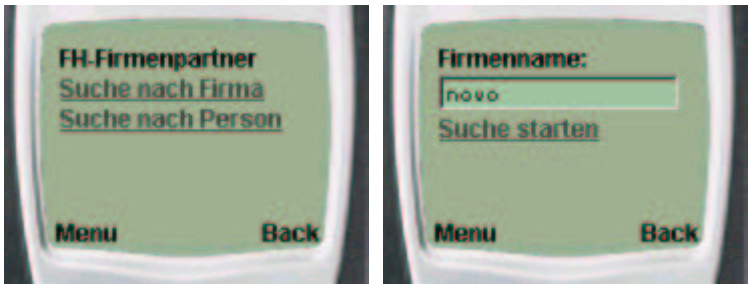


Abbildung 16: Firma suchen

Abbildung 17: Firma anzeigen

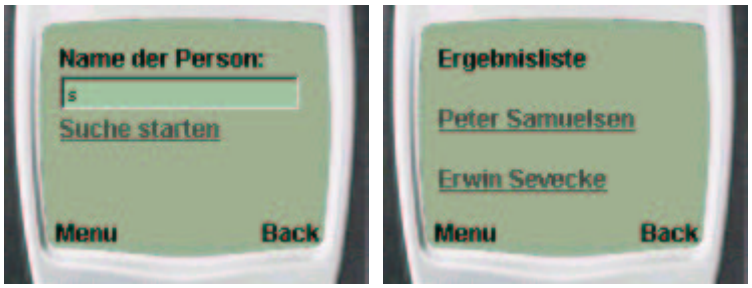


Abbildung 18: Ansprechpartner suchen

Abbildung 19: Ansprechpartner anzeigen

6.3 Realisierungsmodell des Prototyps

Das Diagramm in Abbildung 20 stellt die Konfiguration dar, in der die Hardware- und Softwarekomponenten zum Einsatz kamen.

- Die **Funktionskomponente** des Systems benutzt das TCP/IP Protokoll als Kommunikationsmittel, um auf den *Oracle-Datenbank-Server* zuzugreifen.Die **Interaktionskomponente** stellt folgende Knoten des Systems dar: *GUI Client und Web/WAP-Server*, welche auf der Benutzerseite als Browser und WAP Clients erscheinen. Diese Clients stellen unterschiedliche sprachliche Anforderungen (HTML, WML) an die Web/WAP-Server.
- Die **Entkopplungsschicht** wird durch den *Main-Server* realisiert.

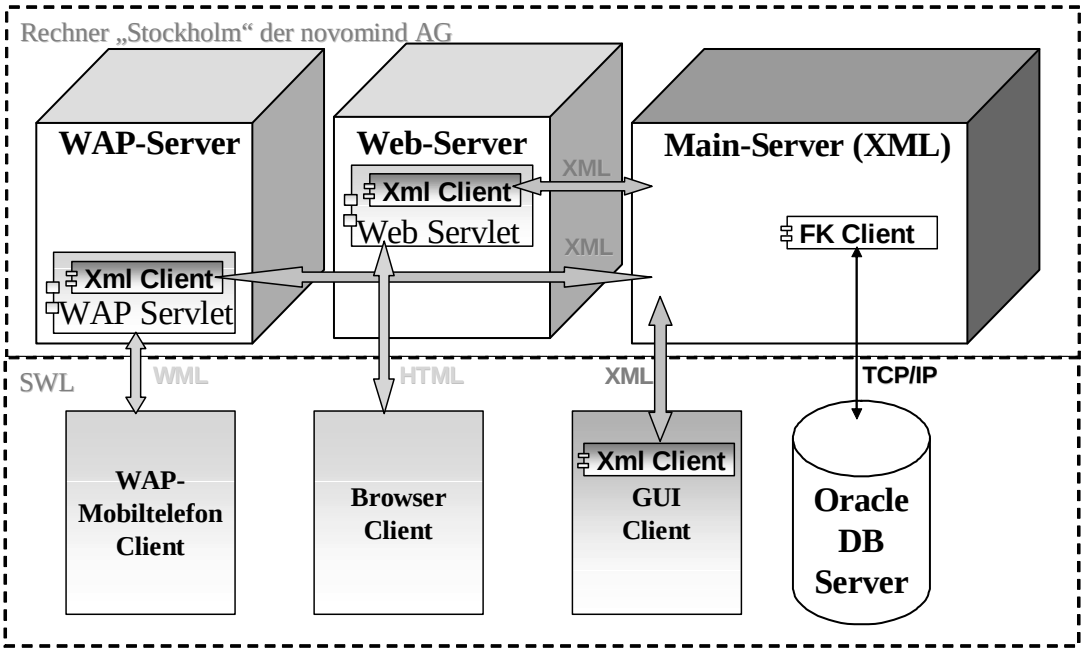


Abbildung 20: Das Realisierungsmodell

Die Clients und die Datenbank befinden sich auf Rechnern des Software-Labors in der FH Hamburg. Eine kurzfristige Implementierung der Server in der FH war aus

Kapazitätsgründen während des Semesters nicht möglich. Daher musste eine andere Möglichkeit außerhalb der FH gefunden werden. Die Firma „novomind AG“ hat uns freundlicherweise für die Dauer des Semesters einen Rechner („Stockholm“) zur Verfügung gestellt.
Aufgrund der streng modularen Architektur unseres Systems war es leicht möglich, die verschiedenen Komponenten, die allein nicht benutzbar sind, auf mehrere Rechner zu verteilen. Das Realisierungsmodell (s. Abbildung 20) lässt erkennen, dass die einzelnen Rechner prinzipiell genauso gut weltweit verteilt sein könnten.

6.4 Erfahrungen mit dem Prototyp

- Wir konnten alle geforderten Schnittstellen realisieren. Während der Implementation wurde deutlich, dass gelegentlich der konzeptionell beste Weg zugunsten einer kurzfristigen Lösung verlassen oder auf einige Funktionalitäten verzichtet werden musste. Wir kamen zu der Erkenntnis, dass an einigen Stellen ein Redesign erforderlich ist.
- Wir hatten die Hoffnung, die im folgenden Szenario wiedergegebene Funktionalität realisieren zu können:

  > Szenario: „Ein Studierender wählt über WAP die Firmensuch-Funktion. Zu seinem Suchkriterium (Beispiel: „novomind“) erhält er zahlreiche Treffer, also verschiedene Abteilungen der Firma mit den jeweiligen Ansprechpartnern und deren Telefonnummern. Der Studierende wählt nun einen Ansprechpartner aus und klickt auf dessen Telefonnummer. *Damit wird die Telefonverbindung aufgebaut*.“

Die Suche konnten wir in der geschilderten Form realisieren, aber nicht den einfachen Aufbau der Telefonverbindung durch Anklicken der Angabe in der WAP-Seite. Die derzeitig verwendete Technologie der Dienstanbieter verfügt nicht über diese Leistung.

- Durch konsequente Modularisierung im Architekturmodell konnten wir während der Realisierung problemlos die Web- und WAP-Komponente auf einen entfernt stehenden Rechner auslagern und damit Installationsprobleme umgehen. Aufgrund der weitgehenden Entkopplung der einzelnen Systemkomponenten wurde die Verteilung vollkommen transparent. Daher konnten Komponenten auch bei den Studierenden zu Hause entwickelt und problemlos in der Hochschule integriert werden.

7. Ergebnis und Ausblick

Aufgrund der geschilderten Projekte konnte ein Beleg dafür erbracht werden, dass einfache, alltägliche Geräte wie ein Organizer (PDA) oder ein Mobiltelefon als Frontend für web-basierte Anwendungen prinzipiell tauglich sind. In der Praxis sind natürlich die sehr kleinen Bildschirmanzeigen sowie die teilweise umständliche Bedienung und die geringen Netzbandbreiten hinderlich für eine breite professionelle Nutzung dieser Geräte etwa in der Bearbeitung von Geschäftsprozessen. Allerdings sind die technologischen Entwicklungen noch lange nicht abgeschlossen, so dass in diesen Problembereichen in den nächsten Jahren gravierende Verbesserungen zu erwarten sind. Hinsichtlich der Leistungsfähigkeit der Übertragungswege kann man auch Fortschritte erhoffen, etwa von UMTS (s. [24]). Der Ansatz, durch alltägliche Geräte den Zugang zu Gemeinschaften zu unterstützen und zu ermöglichen, wird in Projekten an der FH Hamburg weiterhin verfolgt werden.

Die beschriebenen Entwurfsentscheidungen (Nutzung von XML, TCP/IP,...) haben sich ausnahmslos bewährt. Insgesamt entstand eine große Freiheit in der Lokalisation von Systemkomponenten unter Nutzung des Internet.

Das Konzept, im Rahmen von Lehrveranstaltungen mit eher grundlegendem Charakter aktuelle Anwendungskonzepte und Technologien zu thematisieren, erwies sich wiederholt als sinnvoll. Aus der Bearbeitung von Themen, die in der Wirtschaft gegenwärtig intensiv diskutiert werden, resultiert eine zusätzliche Motivation für die Studierenden.

Danksagung

An der Planung und Realisierung der hier besprochenen Projekte waren jeweils etwa 25 Studierende im Software-Engineering-Praktikum des Sommersemesters 2000 (Palm-Data-Warehouse) und des Wintersemesters 2000/2001 (WAP) beteiligt. Allen sei herzlich für ihren Beitrag gedankt.

Besonderer Dank gilt Herrn Samuelsen von der Firma novomind AG für die kontinuierliche Unterstützung unserer Projekte, die unter anderem zur großzügigen Überlassung eines Servers für die Projektdauer geführt hat.

8. Literatur

[1] K. Beck, W. Cunningham: A Laboratory for teaching object-oriented thinking. Proceedings of OOPSLA ’89, pp. 1-6, 1989.

[2] Behme/Mintert: XML in der Praxis; Addison/Wesley, 2000

[3] Document Object Model (DOM): http://www.w3.org/DOM/, 2001

[4] Marcel Ecks, Matthias Senft, Jörg Raasch: Die technische Infrastruktur zur Teilnahme von Unternehmen an Gemeinschaften in Neuen Medien. Workshop GeNeMe99, in [5] pp.67-86

[5] Martin Engelien, Kai Bender (Hrsg.): GeNeMe98, Gemeinschaften in neuen Medien. Tagungsband TU Dresden 1./2.10.1998, Josef Eul Verlag, 1998.

[6] Martin Engelien, Detlef Neumann (Hrsg.): Virtuelle Organisation und Neue Medien 1999. GeNeMe1999, Gemeinschaften in neuen Medien. Tagungsband TU Dresden 28./29.10.1999, Josef Eul Verlag, 1999.

[7] Christiane Floyd: A Systematic Look at Prototyping. In: ed. R. Budde, K. Kuhlenkamp, L. Mathiassen, H. Züllighoven: Approaches to Prototyping. Berlin, Heidelberg, New York, Tokyo: Springer, 1984, S. 1-18.

[8] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns – Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.

[9] WAP Browser @ gelon.net: http://www.gelon.net

[10] Jason Hunter, Brett McLaughlin: Java Servlets & Java und XML. O'Reilly, JAHR???

[11] JDOM: http://www.jdom.org, 2001

[12] Sun microsystems: Java Servlet Development Kit in: Java Servlet Technology http://java.sun.com/products/servlet/

[13] Sun microsystems: Java Servlet Technology; http://java.sun.com/products/servlet/

[14] Peter Köller: Servlets und JavaServer-Pages; Universität Ulm Fakultät Informatik, Abteilung Verteilte SystemeWintersemester 1999/2000 http://www.metaprojekt.de/Programmieren/Java/Servlets/

[15] Glenn E. Krasner, Stephen T. Pope: A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80. JOOP, 1(3), 1988, 26-49.

[16] M. Lusti: Data Warehousing und Data Mining, Springer 1999.

[17] Objectivity Technical Overview, Release 5, August 1999 //www.Objectivity.com/Products/TechOv.html

[18] Oracle 8i Online Generic Documentation Library

[19] Dokumentation und Information über Palm-Organizer (Warenzeichen der Palm Computing Inc.). http://www.palm.com

[20] Jörg Raasch: Komponentenarchitektur für verteilte Systeme. Workshop GeNeMe98, in [6] pp.49-72

[21] SAX 2.0: The Simple API for XML: http://www.megginson.com/SAX/, 2000

[22] mod_ssl: The Apache Interface to OpenSSL: http://www.modssl.org/, 2001

[23] JavaServer Pages(TM) - Tomcat @ Jakarta: http://java.sun.com/products/jsp/tomcat, 2001

[24] UMTS - eine kurze Einführung: http://www.teltarif.de/i/umts.html, 1997 - 2001

[25] Till Valentin: Einsatz von Handheld-Geräten im Versicherungsaußendienst. Diplomarbeit, FH Hamburg, Februar 2001.

[26] Voicebox: Franks Exchange FAQ - Telefon als Client: http://www.msexchangefaq.de/clients/voice.htm, 2000

[27] M. Weiser: Some Computer Science Problems in Ubiquitous Computing. CACM, Juli 1993.

[28] WAP.net - The place to be, for all things WAP! http://www.wap.net/

[29] XML.com: XML From the Inside Out: http://www.xml.com/, 2001
XML.ORG - The XML Industry Portal: http://www.xml.org, 2001
developerWorks : XML: http://www.ibm.com/developer/xml/, 2001
Extensible Markup Language (XML):http://www.w3.org/XML/, 2001

[30] Heinz Züllighoven: Das objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz. Heidelberg: dpunkt-Verlag, 1998