Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Projekt

Alewtina Schumann

Indoornavigationssystem im Flughafenszenario

Fachbereich Elektrotechnik und Informatik
Department of Electrical Engineering and Computer Science

Alewtina Schumann

Thema der Ausarbeitung Projekt

Indoornavigationssystem im Flughafenszenario

Stichworte

Indoornavigation, Profiling, Verteilte Systeme, Mobile Computing, E-Commerce

Kurzzusammenfassung

Dieses Dokument beschreibt Erfahrungen, die im Projekt Indoornavigationssystem in einer Flughafenmetapher gesammelt wurden. Ausgehen von dem Flughafen-Szenario werden die Anforderungen an das Informationssystem analysiert. Es wird die Architektur des gesamten Systems und der Vorschlagskomponente und die Implementierung dieser Komponente sowie der grafischen Oberfläche vorgestellt. Außerdem werden Schwierigkeiten und Probleme beschrieben, die während des Projektes aufgetreten sind.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Motivation

Heutzutage besitzen Flughäfen riesige Flächen und schließen mehrere Gebäuden mit ein. Sie sind zu komplizierten Strukturen geworden mit vielen Verschiedenen Funktionalitäten. Zu den Geschäftsfeldern in einen Flughafen gehören nicht nur Passagieren- Beförderung, sondern auch Tourismus, Hotels, Autovermietungen, Gastronomie, Einzelhandel. Die Terminals und die verschiedenen Geschäfte befinden sich in großen Gebäuden, wo man sich nicht ganz einfach zu Recht findet. In dem Flughafenprojekt soll ein Prototyp für ein Indoor-Navigationssystem am Beispiel des Frankfurter Flughafens entwickelt werden, das es den Flughafengästen ermöglicht, sich auf dem Flughafengelände besser zu orientieren.

Die Flughafengäste sollen auf dem Flughafengelände schnell und bequem zu ihren Points-of-Interest geführt werden. Das Führen heißt aber nicht nur, dass eine Route von Punkt A nach Punkt B berechnet und angezeigt wird. Dem Fluggast werden auf Grund seines Profils und seiner aktuellen Position auf dem Flughafengelände Geschäfte vorgeschlagen, die für ihn interessant sein könnten. Der Frankfurter Flughafen macht die meisten Umsätze mit seinen Geschäften und nicht, wie man es sich vorstellen würde, mit den Flügen selbst. Und es gibt noch mehr Potenzial auf diesem Gebiet. Wenn man die Fluggäste gezielt zu den Geschäften führt, kann man die Umsätze noch mehr steigern. Die Route kann so berechnet werden, dass andere Points-of-Interest mit einbezogen werden (das entscheidet das System mittels Benutzerprofils), auch wenn der Fluggast es nicht explizit dem System befohlen hat. Außerdem kann virtuelle Werbung, die ein Fluggast auf seinem mobilen Gerät erhält, ihn auf eine oder andere Ware aufmerksam machen.

Der Prototyp des Informationssystems besteht aus einer stationären Server- und mobilen Client-Anwendung. Die Positionsbestimmung passiert mittels IMAPS- Modulen [Edyta Kutak], die aus einem Sender und einem Empfänger bestehen. Das Gebäudemodel und die Dienstebeschreibung werden im Indoor Map Server [Jan Napitupulu] festgehalten. Diese Ausarbeitung beschäftigt sich mit dem Client-Teil des Projektes und einer Vorschlagskomponente, die aufgrund des Benutzerprofils [Milen Koychev] dem Flughafengast verschiedene Vorschläge macht und für die intelligente Suche nach Diensten zuständig ist.

2 Analyse

In diesem Kapitel werden die wichtigen funktionalen sowie nicht funktionalen Anforderungen an das System *aus der Sicht der Vorschlagskomponente und der Interaktionskomponente* beschrieben und analysiert. Um diese besser zu verstehen wird zuerst das Projektszenario vorgestellt.

2.1 Projektszenario

Das ursprüngliche Projektszenario, das hier präsentiert wird konnte aus zeitlichen Gründen nicht vollständig implementiert werden. Es dient aber als Basis für die Analyse und das Design der System-Architektur. Nach diesem Szenario

kommt ein Geschäftsmann aus Hamburg mit dem Taxi zum Flughafen und meldet sich mit seinem PDA beim Flughafen-System an. PDA teilt ihm mit, dass aufgrund einer Reparaturmaßnahme sein Flug nach Genf mindestens 3 Stunden Verspätung hat. Mit Hilfe von PDA setzt der Geschäftsmann seine Geschäftspartner in Genf über die Verspätung in Kenntnis und benachrichtigt den Abholservice. Er versucht die Zeit zu nutzten und sucht nach einer für ihn interessanten Veranstaltung. Hunger hat er auch noch. Und tatsächlich läuft im Flughafen eine Messe, die er besuchen kann. Außerdem schlägt der PDA 3 Restaurants, die auf dem Weg zur Messe liegen und dem Profil (dem Geschmack) des Geschäftsmanns entsprechen. Er entscheidet sich für italienische Küche und PDA reserviert einen Tisch. Auf dem Weg zum Restaurant, bekommt er über seinen PDA profilbezogene Werbung. Ein Rollex-Angebot ist sehr verlockend und er will sich die Uhr angucken. So weicht er von der Route ab. Der PDA zeigt die neue Route an, storniert die alte Tischreservierung und bucht einen neuen Tisch für spätere Uhrzeit.

Auf der Messe, wo er glücklich angekommen ist und schon einige Sachen angeschaut hat, bekommt der Geschäftsmann die Nachricht mit der genauen Abflugzeit, da die Reparaturen abgeschlossen sind. Von der Messe geht er zu seinem Terminal. Da die Abflugzeit und damit auch die Ankunftszeit bekannt sind, teilt der PDA den Geschäftspartnern in Genf das mit, und bestellt gleich ein Taxi in Genf.

In Flugzeug will der Geschäftsmann einige Verträge und eine Präsentation anschauen. Er überträgt die Sachen von seinem PDA auf ein im Sitzt eingebautes Bildschirm, um eine bessere Ansicht auf die Dateien zu bekommen.

2.2 Funktionale Anforderungen

Im Folgenden werden die wichtigen Funktionalitäten und Anwendungsfälle dargestellt. Es soll dem Benutzer möglich sein, sich beim System anzumelden, nach den Diensten suchen, die er später in Anspruch nehmen kann. Das System soll auch unabhängig von den Benutzerbefehlen aufgrund des Benutzerprofils und seiner aktuellen Position geeignete Vorschläge machen und evtl. virtuelle Werbung der Dienste an den Benutzer weiterleiten. Wenn der Benutzer einen Dienst in Anspruch genommen hat, wird das durch das System in sein Profil eingepflegt. Die Abbildung 2.1 zeigt die Funktionalitäten des Systems, die für die Dienstsuche und Benutzen von Diensten verantwortlich sind.

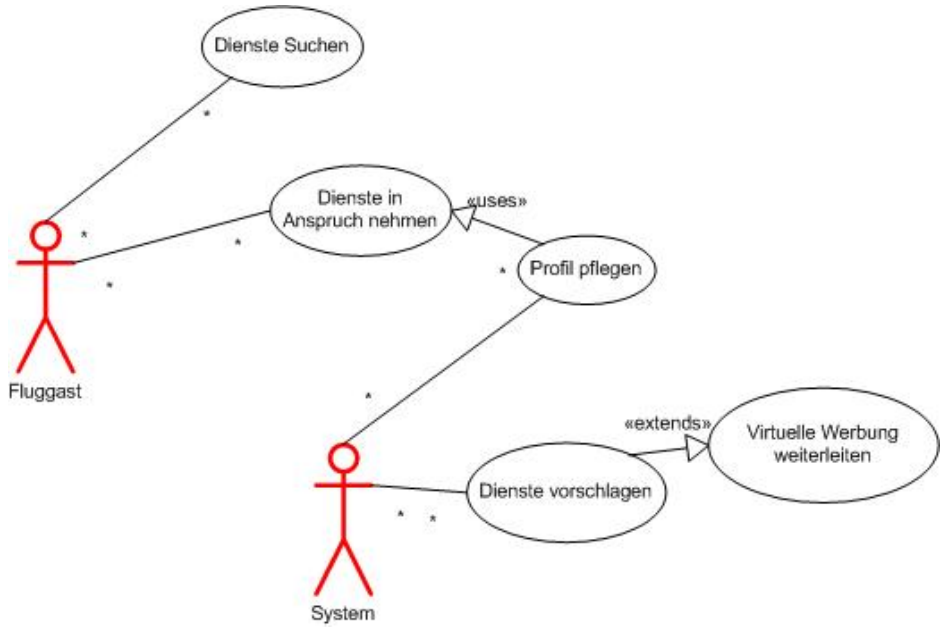

Abbildung 2.1: Use Case: Dienste suchen und vorschlagen

Außerdem gibt es Navigationskomponente, die eine geeignete Route zu dem vom Flughafengast gewählten Dienst berechnet und anzeigt, und für das Tracking der Person verantwortlich ist. Die Funktionalitäten dieser Komponente sind in der Abbildung 2.2 dargestellt.

Nach dem ein den Benutzer interessierter Dienst gefunden wurde, berechntet das System die Route von der aktuellen Position der Person zu diesem Dients. Der Algoritmus dafür

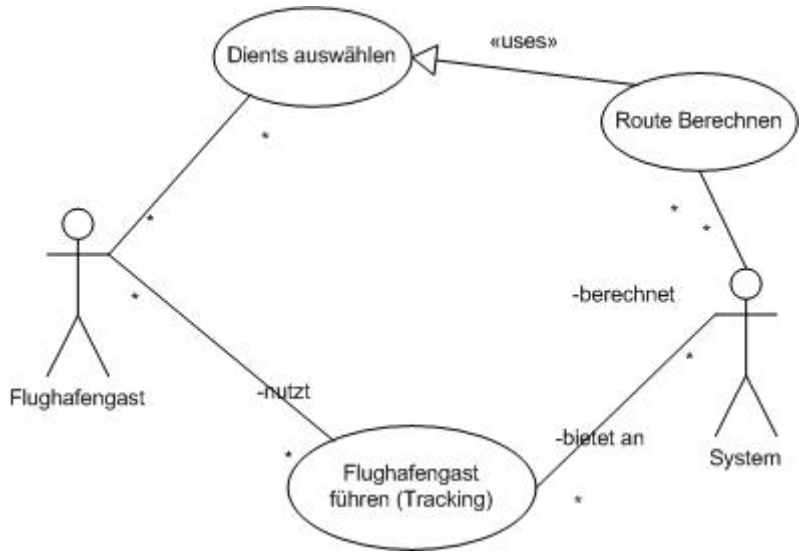

Abbildung 2.2: Use Cases: Navigation

könnte die Profildaten benutzen, um bessere Route zu erstellen, die Dienste miteinbeziet, welche für die navigierte Person interessant sein können.

2.3 Nicht funktionale Anforderungen

Außer Anforderungen, die die Funktionalität der Software beschreiben und in kommerziellen Projekten von Kunden gestellt werden, gibt es weitere, nichtfunktionale Anforderungen. Diese Anforderungen stellen die Qualitätsmerkmale des Softwaresystems dar. Sie beschreiben wie ein System die funktionalen Anforderungen erfüllen soll. Diese Anforderungen bringen oft die Komplexität ins Projekt und verursachen dadurch auch die höchsten Kosten.

Da das System, genauer sein Client-Teil, unter anderem auf unterschiedlichen mobilen Geräten laufen soll, interessieren uns besonders solche Aspekte wie Portierbarkeit(Lauffähigkeit auf unterschiedlichen Plattformen) und Ergonomie. Unterschiedliche mobile Geräte haben verschiedene kleiner als gewöhnliche Bildschirmgrößen, Tastenanzahl, die kleiner ist als die von der normalen Tastatur. Sie laufen unter unterschiedlichen Betriebssystemen. Das alles muss bei dem Entwurf berücksichtigt werden und das System soll sich an seine Umgebung anpassen können. Das System ist stark benutzerorientiert und muss deswegen bequem und unkompliziert in der Nutzung sein. Daher soll die grafische Oberfläche so gestaltet werden, dass der Benutzer keine Schwierigkeiten bei der Nutzung des Systems hat. Sicherheit ist auch ein relevanter Punkt, denn es wird mit persönlichen Daten gearbeitet und anhand die-

ser werden Profile erstellt. Der Systemnutzer soll sicher sein, dass seine persönlichen Daten geschützt sind und nicht missbraucht werden.

3 Architektur

Aus der Analyse hat sich ergeben, dass das System auf eine Client-Server-Architektur basiert. Da mobile Geräte über eine sehr eingeschränkte Speicherkapazität verfügen, soll die Client-Anwendung möglichst schlank sein und nicht viel mehr als eine Interaktionskomponente enthalten. Außerdem soll der IMAPS-Empfänger hardware- sowie softwaremäßig auch auf dem Client laufen und die aktuellen Benutzerkoordinaten an den Server senden. Auf dem Server hat sich die Geschäftslogik des Systems angesiedelt, die für Profilerstellung und -pflege, Vorschläge von Diensten und Personenführung zuständig ist. Das Gebäudemodel und die Dienstebeschreibung befinden sich auch auf der Serverseite. Die Abbildung 3.1 zeigt die Gesamtarchitektur des Systems.

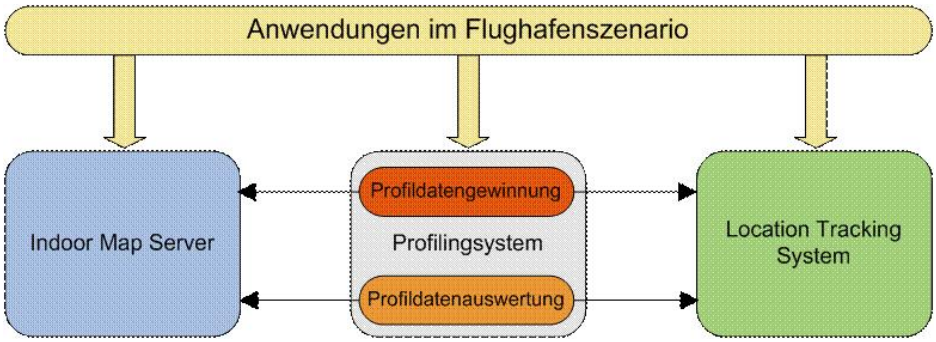

Abbildung 3.1: Gesamtarchitektur

In dieser Abbildung ist der Clientteil sowie sie Vorschlafskomponente als „Anwendungen im Flughafenszenario“ bezeichnet. Hier sieht man aber schon die Profile-Komponente, die eine Basis für die Vorschlagskomponente darstellt. Die Profil-Komponente nutzt Informationen aus *Indoor Map Server* und *Location Tracking System* um die Profildaten zu gewinnen. Die Daten werden in einer Datenbank abgespeichert und verwaltet.

Die genaue Betrachtung der Vorschlagskomponente präsentiert die Abbildung 3.2

Auf dem Server läuft die Suggestion-Logik. Sie wertet Informationen aus, die sie von IMAPS und Profile-History erhält, um für den Benutzer geeignete Vorschläge zu machen. Wie das genau passiert wird im Kapitel „Implementierung“ beschrieben. Da die Profile-Komponente ihre eigene Dienstebeschreibung hat, die sich von der LBS-Beschreibung unterscheidet,

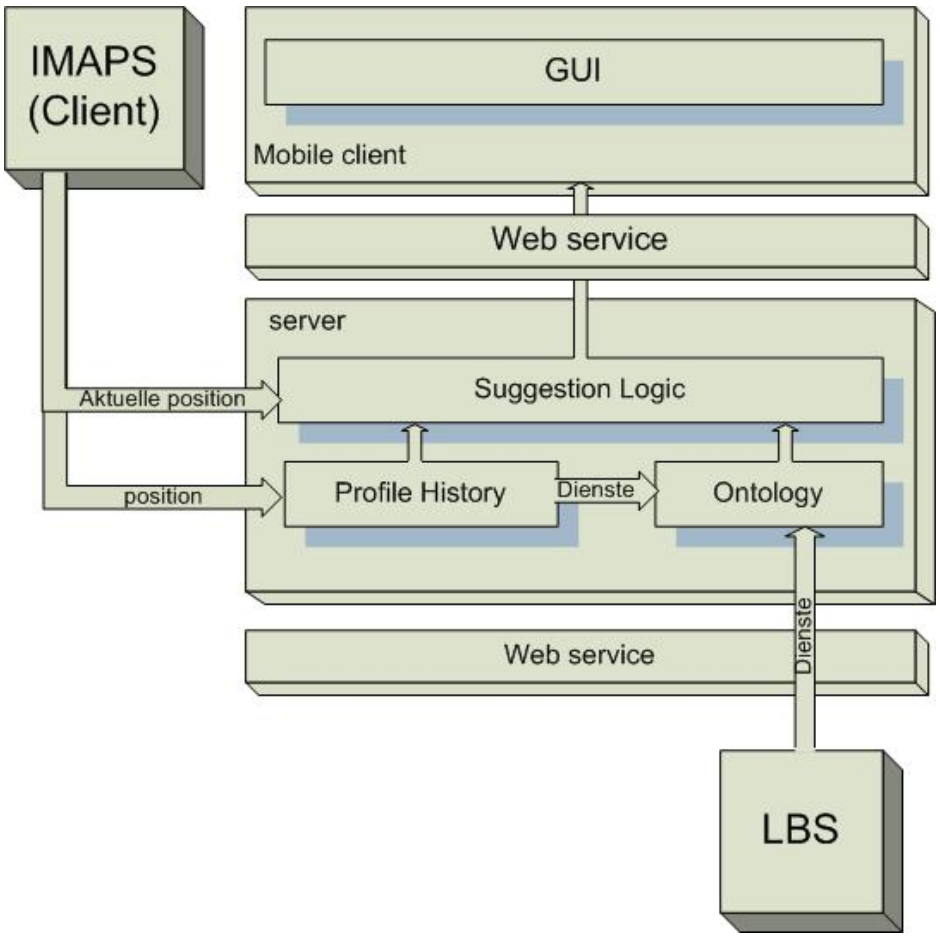

Abbildung 3.2: Architektur: Vorschlagskomponente

die Vorschlagskomponente aber diese beiden Beschreibungen für ihre Auswertungen nutzt, müssen sie verglichen werden können. Das macht die Ontology-Komponente, die die Diensteontology beschreibt. Nachdem die Suggestion-Logik passende Dienste gefunden hat, leitet sie ihre Ergebnisse an die Interaktionskomponente auf dem mobilen Client. Die Kommunikation zwischen allen Komponenten (Client, Server, LBS) läuft über Web Service.

4 Implementierung

Die im vorigen Kapitel vorgestellte Architektur soll nun in einem Prototyp implementiert werden. Dieses Kapitel beleuchtet die Einzelheiten, die bei der Implementierung beachtet wurden. Des Weiterem wird erläutert wie die Designvorgaben umgesetzt werden und die Realisierung der einzelnen Komponenten der Suggestion-Logik und der Client-Seite vorgestellt.

4.1 Implementierungstechnologie

Für die Realisierung des Prototyps wurde MS .Net Technologie ausgewählt. Zwar hatten die meisten Mitglieder des Entwicklerteams keine oder wenig Kenntnisse über diese Technologie, bietet .Net aber viel bessere Werkzeuge als z.B. Java, besonders auf dem Gebiet der Softwareentwicklung für mobile Geräte und der Entwicklung der grafischen Benutzeroberfläche. Der Einarbeitungsaufwand in diese Werkzeuge und Programmiersprache .Net C# und .Net Compact Framework für die Entwicklung für mobile Geräte war deutlich geringer, als der Aufwand, den man für die Entwicklung des Systems unter Java oder ähnlichen Sprachen notwendig wäre. Ein weiterer Vorteil von der .Net Technologie war es, dass die ganze Software, die für die Entwicklung des Systems notwendig war wie z.B. Datenbankserver, Entwicklungsumgebung, .Net Framework u.a., wurden dem Entwicklerteam durch MSDNAA-Programm kostenlos zu Verfügung gestellt. Außerdem hatte das Team eine große Bereitschaft für das Lernen und Einsetzen der neuen Technologie erwiesen.

4.2 Einzelne Komponenten

Das Komponentendiagramm, das in der Abbildung dargestellt ist, zeigt alle Komponenten des Prototyps. Diese Arbeit beschäftigt sich mit der Ontology, Suggestionkomponente und Interaktion.

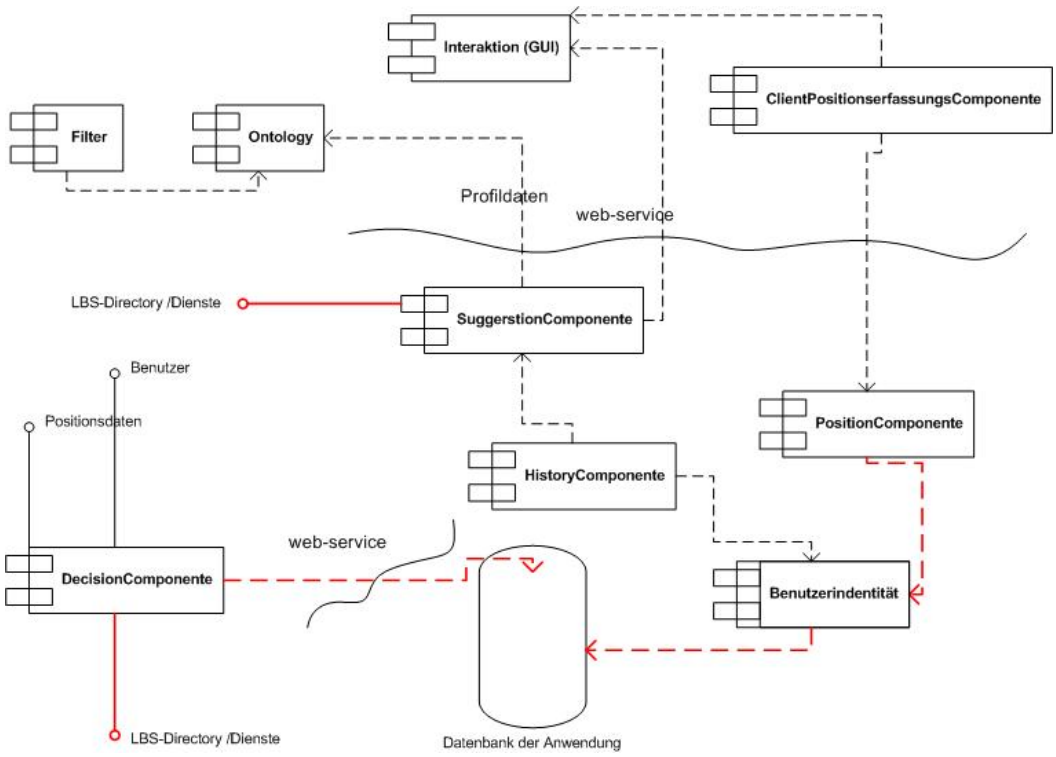

Abbildung 4.1: Komponentendiagramm

4.2.1 Ontology

Die Profilkomponente sammelt Informationen, wann ein Benutzer, der den Flughafen besucht, welche Dienste in Anspruch genommen hat. Diese Informationen speichert sie als Tabellen in eine relationale Datenbank. Also werden die einzelnen Dienste und deren Eigenschaften auch in Tabellen abgebildet. Der LBS hat aber sein eigenes Model für die Beschreibung der Dienste. Die Vorschlagskomponente nutzt die Daten der Profilkomponente, um zu entscheiden, welche Dienste dem Benutzer interessant sein könnten, und diese ihm dann vorzuschlagen. Die LBS-Daten nutzt sie auch, um zu finden, wo sich die evtl. interessante Diente, die sie entsprechend dem Benutzerprofil ermittelt hat, im Flughafen befinden und ob sie sich entsprechend nah dem Flughafengast liegen. Also muss die Vorschlagskomponente die Dienstebeschreibung beider anderen Komponenten vergleichen können, um dann ein endgültiges Ergebnis zu erstellen. Die Ontology beschreibt die Welt der Dienste, die in einem Flughafen angeboten werden, und sie dann vergleichen zu können. Die Ontology wir durch ein Prolog-Programm dargestellt, wo die Dienste beschrieben werden. Ein Prolog-Code unten zeigt an einem Beispiel, wie die Beschreibung der Dienste aussieht. Solcher Programm-Code wird dann in C# mit Hilfe der SWI-cs-Komponente eingebettet, so dass man Anfragen an die Ontology direkt über C# stellen kann.

```
italien(Luigis'Pizza).
chineese(Pekin).
spanish(Ola!).
mediterran(name) :- italienish(name)| spanish(name).
asian(name) :- chineese(name).
essen(name) :- mediterran(name) | asian(name).
```

4.2.2 Suggestionkomponente

Die Ontology ist eigentlich nur ein Teil der Suggestion- (oder Vorschlags-) Komponente. Die Suggestionkomponente bereitet die Daten für sie vor und nutzt die Ergebnisse der Ontologykomponente für weitere Auswertung. Sie sucht anhand der aktuellen Position des Benutzers und seines Profils nach Diensten, die für den Benutzer nützlich und interessant sein können. Sie dient auch als intelligenter Filter, wenn der Flughafengast explizit eine Suche nach einem Service gestartet hat. Die Suggestionkomponente zeigt die Dienste als Ergebnis an, die z.B. auf dem Weg zum Abfluggate der Person liegen, oder die der Benutzer bevorzugt (z.B. mediterrane Küche statt asiatischer).

Aufgrund des Zeitmangels nutzt diese Komponente einfache Algorithmen, um zu entscheiden, welcher Service für den Vorschlag geeignet ist. Mit mehr Zeit wäre die Profilkomponente ausgereifter, so dass man auch die Vorschlagskomponente mit besseren Algorithmen ausstatten könnte. Man könnte z.B. in Richtung „Data Mining“ gehen und mehr Informationen den Profilen entnehmen. Außerdem könnte die Suggestionkomponente die virtuelle Werbung empfangen, die nützlichen Informationen rausfiltern und an den Benutzer weiterleiten. Im Moment ist nur eine Schnittstelle für die virtuelle Werbung vorgesehen und implementiert.

4.2.3 Interaktionskomponente

MS Visual Studio 2005 bietet sehr gute Werkzeuge für die Entwicklung der grafischen Benutzungsoberflächen, mobile Geräte nicht ausgenommen. Trotzdem musste man sich auf die kleine Bildschirmgröße umstellen. Mobile Geräte schränken Entwickler ein, das sie nicht so viele Möglichkeiten anbieten, wie die gewöhnlichen Rechner. Sie haben weniger Tasten, man kann nicht alle grafischen Komponenten für die Realisierung der Benutzungsoberfläche einsetzen. Außerdem haben die weniger Speicher und eingeschränkte Akku-Laufzeit. Das alles muss man berücksichtigen und trotzdem eine benutzerfreundliche Interaktionskomponente implementieren.

4.2.4 Kommuntikation zwischen den Komponenten

Die Anwendung teilt sich wie schon früher gesagt in einen mobilen Client und einen Server-Teil auf. Die Kommunikation zwischen den beiden passiert über WebServices-Technologie. Auch einige Server-Komponenten, die sich auf unterschiedlichen Servern befinden, benutzen WebServices für die Kommunikation. So erhalten die Suggestion- und Profilkomponente Informationen von LBS nur über WebService.

Eine andere Kommunikation geschieht zwischen IMAPS-Empfänger, der die aktuellen Koordinaten einer Person liefert, und dem System. Die Software für IMAPS ist in Java realisiert. Das ganze System ist jedoch in .Net C# geschrieben. Profilekomponente sowie Suggestion brauchen die Informationen über die aktuelle Position des Benutzers, um ihren Aufgabenteil zu erfüllen. Als Lösung würde die Interprozesskommunikation eingesetzt. Hat IMAPS-Empfänger neue Daten bekommen, sendet er diese über einen Socket. Der C#-Teil startet seinerseits einen Prozess, der auch über einen Socket die Daten empfängt. Diese Lösung ist nicht optimal, und nur für den Prototyp entwickelt. Für die weitere Entwicklung des Systems sollte man IMAPS-Empfänger umprogrammieren und auf .Net Technologie setzen.

5 Probleme und Fazit

In diesem Kapitel werden Probleme beschrieben, die während des Projekts aufgetreten sind. Ein kurzes Fazit dient als Abschluss der Ausarbeitung.

5.1 Probleme

Das größte Problem war die Zeit, an der fast bei jedem Hochschulprojekt mangelt. Am Anfang ging viel Zeit (ca. 3 Wochen) verloren, indem der Versuchsraum eingerichtet war, das Seilsystem mit den IMAPS-Sendern installiert (eingeschlossen das Löten der IMAPS-Crickets) und der Raum gemessen wurde. Zwar waren diese Maßnahmen notwendig, wurden aber nicht oder mit weniger Aufwand ins Projekt eingeplant. Die Einarbeitung in die .Net Technologie wurde vor dem Projektbeginn vorgenommen und hat keine großen Schwierigkeiten während des Projekts bereitet. Trotzdem hat .Net Compact mehrere Einschränkungen, die für das standardmäßige .Net Framework nicht gelten und viele Sachen mussten mehrmals modifiziert werden, bevor sie funktioniert haben. Wie z.B. Funktionalität für den WebService, die die Benutzer-Authentifikation automatisch mit sich trägt und die im Prototyp zuerst eingesetzt wurde. Diese Funktionalität gibt es im Compact Framework nicht, so dass das System auf mobilen Geräten nicht lief. So musste man schließend ein eigenes Authentifikationsmodul entwickeln.

Ein anderes Problem war die Abhängigkeit der Vorschlagskomponente von den Daten, die Profilhistory und LBS liefern. Solange diese beiden Komponenten modelliert wurden und keine Schnittstellen vorhanden waren, konnte die Suggestionkomponente sehr langsam entwickelt werden.

Große Schwierigkeiten hatte das Team mit der Positionsbestimmung-Hardware. Die IMAPS-Crickets sind sehr unstabil und müssen noch verbessert werden. Sie liefern noch viele ungenaue und sogar falsche Daten. Das liegt zum Teil daran, dass der Versuchsraum in virtuelle Räume aufgeteilt war und keine richtigen Trennwände existierten, sodass der Empfänger auch von weit entfernten Sendern Signale bekommen hat. Auch der Versuch die Software des Empfängers in C# umzuschreiben, der einige Zeit des gesamten Teams in Anspruch genommen hat, war gescheitert.

5.2 Fazit

Trotz allen Problemen im Projekt wurde ein lauffähiges Prototyp erstellt, der nach Diensten sucht, die dem Benutzerprofilentsprechen, eine Route zu dem gewählten Dienst anzeigt und die Person zu diesem Dienst führt (tracking). Das Projekt sollte weitergeführt und alle Komponenten in einer Iteration verbessert werden. Aus der Sicht der Vorschlagskomponente könnte man weiter gehen und „Data Mining“ für Profilauswertung einsetzen. Das wäre aber nur dann möglich, wenn einige Änderungen an der Profilkomponente stattfinden. Die Suggestion könnte auch zu einem privaten Entscheidungsassistenten erweitert werden, der dem Benutzer hilft seine Aufenthaltszeit am Flughafen besser zu gestalten.