Bachelorarbeit

Kristoffer A. Witt

Eine verteilte Anwendung zur Nutzung von Ortsbasierten Diensten mit mobilen Endgeräten

Fachbereich Elektrotechnik und Informatik
Department of Electrical Engineering and Computer Science

Kristoffer A. Witt

Eine verteilte Anwendung zur Nutzung von Ortsbasierten Diensten mit mobilen Endgeräten

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
Studienrichtung Softwaretechnik
am Fachbereich Elektrotechnik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Gunter Klemke
Zweitgutachter : Prof. Dr. Kai von Luck

Abgegeben am 17. August 2008

Thema der Bachelorarbeit

Eine verteilte Anwendung zur Nutzung von Ortsbasierten Diensten mit mobilen Endgeräten

Stichworte

J2ME, Bluetooth, Ortsbasierte Dienste, Location Based Service, mobile Endgeräte, Mobiltelefon

Kurzzusammenfassung

Design und Implementierung einer in Client und Server gegliederten Anwendung zur Nutzung von Ortsbasierten Diensten mit Mobiltelefonen.

Title of the paper

A distributed application for accessing location based services with mobile terminals.

Keywords

J2ME, Bluetooth, Location Based Service, mobile terminals, cellphones

Abstract

Design and implementation of a distributed application to exploit location based services with mobile phones.

Danksagung

An dieser Stelle möchte ich folgenden Person für die moralische und beratende Unterstützung bei dieser Arbeit danken.

Prof. Dr. Gunter Klemke und Prof. Dr. Kai von Luck Danke für die gute Betreuung und die Ratschläge.

Meinen Freunden und meinen Eltern Für das ständige nachhaken wann ich den endlich fertig bin.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einleitung

Im Rahmen des Tages der offenen Tür öffnet der Fachbereich Informatik der Hochschule für angewandte Wissenschaft Hamburg (HAW Hamburg, 2008) seine Türen interessierten Besuchern.

Am Eingang erwartet die Besucher eine Installation mit folgender Aufschrift:

Schön das Sie uns besuchen. Besitzen Sie ein Mobiltelefon der neueren Generation? Dann haben wir für Sie ein Angebot, dass Ihren Besuch noch kurzweiliger und interessanter gestalten wird! Unsere Räumlichkeiten sind ausgestattet mit einem Informations- und Interaktionssystem das es Ihnen ermöglicht, noch mehr Informationen zu bekommen, aktiv an unseren Projekten teilzuhaben und uns Ihre Meinung kundzutun. Sprechen Sie einfach einen unserer Mitarbeiter an, er wird Ihnen bei der Installation und Einrichtung behilflich sein. Es dauert nicht lange, versprochen.

Klingt noch ein wenig abstrakt? Dann werden wir mal konkreter. Mit der Software haben sie die Möglichkeit, je nachdem welches Projekt Sie gerade besuchen faszinierende Dinge zu tun und zu erleben! Immer wenn sich eine Möglichkeit bietet, etwas auszuprobieren oder Informationen zu bekommen werden Sie automatisch entweder per Vibration oder per Audio Signal informiert. Wie, dass können natürlich Sie entscheiden!

Hier mal einige Beispiele:

Projekt Roboterfußball:

Haben Sie sich schon einmal gefragt, warum tut ein Roboter das was er tut? Warum fährt er mit rasanter Geschwindigkeit im Kreis? Woher weiß er wo das Tor steht und welches das richtige ist? Wenn Ihre Antwort „Ja“ lautet haben Sie heute die Möglichkeit diese Fragen beantwortet zu bekommen. Über Ihr Mobiltelefon bekommen Sie live im Spielbetrieb aufbereitete Telemetrie-Informationen über die Entscheidungen der Roboter. Warum dreht er nach rechts, warum sucht er jetzt den Ball, alles das lässt sich nun nachvollziehen.

Sollte Ihnen das noch nicht interaktiv genug sein, steuern Sie doch einfach mal, was der Robot macht. Greifen Sie live in das Spielgeschehen ein. Messen Sie sich mit den dreifachen Gewinnern des Inter-FH-Pokals, das klassische Duell Mensch gegen Maschine.

Projekt Allgegenwärtiger Computer:

Haben Sie zu Hause auch Massen an Fernbedienungen für alle Möglichen technischen Geräte? Probieren Sie in diesem Projekt aus, wie es ist, alle Geräte nur mit Ihrem Mobiltelefon zu steuern. Je nachdem wo Sie sich befinden bekommen Sie zusätzlich weitere Informationen geliefert, z.B. Rezeptideen in der Küche oder das Fernsehprogramm im Wohnzimmer. Kommen Sie vorbei und versuchen Sie es!

Für alle Projekte:

Hat Ihnen etwas besonders gut oder schlecht gefallen? Wir freuen uns auf Ihr Feedback! In allen Projekten haben Sie über den Menüpunkt Feedback die Möglichkeit uns Ihren Eindruck zu vermitteln, das Projekt in verschiedenen Kategorien zu bewerten und Nachrichten für den Ersteller zu verfassen.

Weiterhin stehen für Sie eine Anzahl Informations-Ticker bereit. Die Sie, wenn gewünscht, in jedem Projekt ständig über ausgewählte Themengebiete auf dem laufenden halten. Bundesligaergebnisse, Börsenkurse oder aktuelle Wetterinformationen gibt es etwas Neues erfahren Sie es sofort!

Wieder zu Hause und noch nicht genug informiert?

Wenn Sie einwilligen, erstellen wir für Sie ein Profil mit den von Ihnen besuchten Projekten und stellen dann ein Infopaket zusammen, das Sie über unsere Webseite abrufen können. Speziell auf Ihre Interessen zugeschnitten.

Wir hoffen auch für Sie ist etwas dabei, probieren Sie es aus. Sie werden erstaunt sein, wie vielseitig unser Angebot ist! Viel Spaß nun, wünscht das Team des Fachbereichs Informatik der HAW-Hamburg.

1.1 Motivation

Die Welt ist mobil. Allein in Deutschland besaßen laut Aussage des Statistischen Bundesamtes 2006 über 80% der Bevölkerung ein Mobiltelefon (Statistisches Bundesamt, 2007b). Wobei Mobiltelefon ein Begriff ist, der den aktuellen Multifunktionalen-Kommunikations-Zentralen kaum gerecht wird. MP3-Abspielgerät, hochauflösende Digitalkamera, Internetbrowser oder auch Bluetooth Kurzstreckenfunk, alles dies und mehr vereint ein modernes Handy heutzutage in sich.

Diese Vielfalt an Funktion ist es, was ein Mobiltelefon so interessant macht. Obwohl die Tendenz vom Telekommunikationsgerät zur All-In-One-Lösung keine neue ist, gibt es, von den offensichtlichen abgesehen (z.B. Telefonieren), bisher nur wenige Anwendungen mit

sehr eingeschränkter Interaktion (vgl. Abbildung 1.1). Dabei bietet insbesondere der Kurzstreckenfunk und die daraus implizit ableitbaren Positionsinformationen ein Gros an Anwendungsmöglichkeiten: Die Position ist der entscheidende Parameter für die Nutzung von ortsbasierten Diensten. Diese Dienste, Fachbegriff „Location Based Service“, vermitteln dem geneigten Anwender zu seiner aktuellen Position Informationen und Dienstleistungen. Seien es Informationen über Sehenswürdigkeiten einer Stadt die ein Tourist gerade bereist, oder im kleineren Kontext die Speisekarte eines Restaurants vor dem er sich gerade befindet. Die Möglichkeiten sind schier endlos.

Die Kombination von ortsbasierten Dienstleistungen mit der sich ständig verbessernden Ein- und Ausgabekapazität eines Mobiltelefons birgt ein massives Potential. Anwendungsmöglichkeiten kommerziell wie privat sind schnell gefunden. Beispielsweise, eine vernetzte Infrastruktur vorausgesetzt, ließen sich auf der CeBIT, die besuchten Stände speichern um später Zusatzinformationen am Ausgang per USB-Stick-Werbegeschenk zu erhalten und so eine optimale Nachbereitung zu garantieren. Telefonnummern, Kontaktdaten und Produktinformationen könnten in einem virtuellen Rundgang für den Nutzer bereitgestellt werden und ihm so wertvolle Networkingressourcen liefern. Im Privatbereich hingegen ließe sich das Mobiltelefon als eine Art Universalfernbedienung einsetzen. Befindet man sich im Wohnzimmer, steuert man Stereoanlage oder Heimkinosystem und bekommt zu der aktuellen im TV gezeigten Sendung Informationen gezeigt. Im gesamten Haus ließe sich Licht an und ausschalten oder die Heizung steuern. Für das Haus der Zukunft ein optimaler Begleiter. Ein Weg dieses bisher nicht genutzte Potential auszunutzen wird in dieser Arbeit vorgestellt.

Abbildung 1.1: Mundsburgcenter – Ortsbasierter Push-Dienst

1.2 Zielsetzung

Das grundsätzliche Ziel dieser Arbeit besteht darin, eine verteilte Anwendung für Mobile- (Client) und Stationärgeräte (Server) zu entwickeln, die es ermöglicht Positions- bzw. Orts-Kontext-Informationen auszuwerten um Ortsabhängige Dienste anzubieten oder zu Nutzen. Die Kommunikation zwischen Client und Server soll dabei drahtlos erfolgen, um eine größtmögliche Bewegungsfreiheit für den Client zu erreichen und dem Begriff „Mobil“ gerecht zu werden. Das Infrastrukturdesign soll nach Möglichkeit nur geringe Kosten verursachen, d.h. auf eine umfangreiche Positionierungsarchitektur, beispielsweise durch mehrere Sender pro zu überwachendem Quadratmeter, soll zu Gunsten einer angemessenen Positionierungslösung verzichtet werden.

Um von vornherein eine Vielzahl von Client-/Servergeräten bedienen zu können und die Implementierungskosten zu senken, soll auf kostenintensive Zusatz-Hardware verzichtet werden. Das Programm soll auf einer vorher festgelegten Konfiguration von Hard- und Software lauffähig sein. Die dafür benötigten Hardwarekomponenten sollen nach Möglichkeit bereits weit verbreitet und günstig in der Anschaffung sein, um die Anwendung einer Vielzahl von Nutzern anbieten zu können.

1.2.1 Client

Der Client bekommt die Möglichkeit, über ein Eingabemenü mit dem Server zu interagieren um zum Beispiel Zusatzinformationen zum lokalen Kontext abzufragen oder auch auf einem externen Anzeigegerät ausgegebene Darstellungen zu beeinflussen. Dabei ist zu beachten, dass die Darstellung auf dem Client die jeweilig Bestmögliche sein soll. Genauer bedeutet dies, wenn der Client die Möglichkeit der Anzeige von Grafiken besitzt sollte diese Fähigkeit ohne große Umstände Verwendung finden können. Um dies zu erreichen wird ein Darstellungsprotokoll entwickelt, das dem Anforderungsprofil entspricht.

1.2.2 Server

Der Server stellt den Clients Inhalte zur Verfügung, die je nach Standort variieren. Zusätzlich zu der reinen Bereitstellung von Daten soll direkte Interaktion mit dem Server möglich sein. Das bedeutet der Client beeinflusst direkt und andauernd das Angebot des Servers. Als Beispiel könnte ein Spiel angeführt werden, in dem der Client einen Avatar unmittelbar steuern kann.

1.2.3 Standardanforderungen

Die zu entwickelnde Anwendung soll dabei die Hauptanforderungen der Softwaretechnik an eine Qualitativ hochwertige Architektur erfüllen (Bass u. a., 2003):

- Verfügbarkeit (availiability)
- Änderbarkeit / Erweiterbarkeit (modifiability, extendability)
- Performance / Skalierbarkeit (scalability)
- Sicherheit (security)
- Benutzbarkeit (usability)
- Portabilität (portability)

1.2.4 Abgrenzung

Es ist zu beachten, dass die im Rahmen dieser Arbeit entstehende Anwendung nur stellvertretend für eine Vielzahl von Einsatzmöglichkeiten steht. Das bedeutet, der Fokus bei dem Design der Komponenten liegt auf der Maximierung der Wiederverwendbarkeit. Die Beispielimplementierung hingegen stellt nur eine Mögliche Anwendung dar. Diese wurde unter Berücksichtigung der zur Verfügung stehenden Zeit und Komponenten entworfen um die Basisfunktionalität demonstrieren zu können.

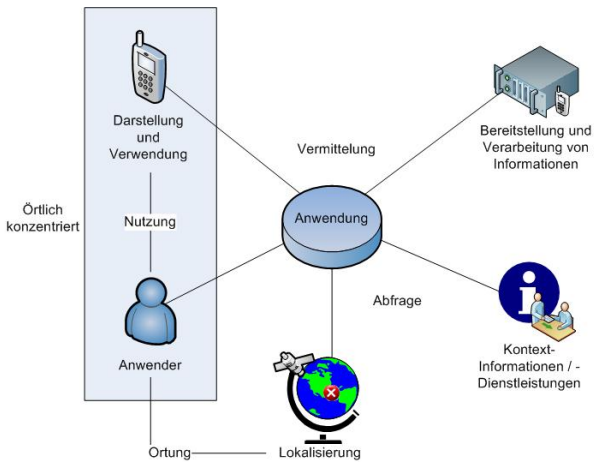


Abbildung 1.2: Anwendungskonzept

2 Verwandte Arbeiten / Marktanalyse

Im folgenden Abschnitt werden zwei sich bereits auf dem Markt oder in der Entwicklung befindende Projekte vorgestellt. Diese Projekte sind entweder vom Aufbau oder vom Leistungsumfang ähnlich zu der von dieser Arbeit angestrebten Lösung. Das Kapitel dient dem Zweck, dem Leser Vergleichswerte zu geben und Unterschiede, Gemeinsamkeiten und Neuerungen besser verdeutlichen zu können.

2.1 UbiZoo

UbiZoo (Keles u. a., 2008) ist ein im Rahmen des Microsoft Imagine Cup 2007 (MIC,(Microsoft, 2007)), entstandenes Projekt, das durch Nutzung von Location Based Services didaktische Unterstützung eines Zoo-Besuchs bietet.

Erstellt und erdacht wurde das Projekt von den HAW-Studenten Eike Falkenberg, Fatih Keles, Jan Napitupulu und Thomas Schmidt mit Unterstützung durch Professor Dr. Olaf Zukunft. UbiZoo hat den deutschen Vorentscheid des MICs gewonnen und durfte somit am internationalen Wettbewerb teilnehmen, dort konnte aber keine der vorderen Platzierungen erreicht werden.

2.1.1 Gliederung

Die Durchführung eines UbiZoo Projekts gliedert sich in drei Phasen:

UbiPrepare – In dieser Phase wird der Besuch geplant, das bedeutet Points-Of-Interest (POI) auf einer Übersicht per GPS festgelegt, zusätzliches Informationsmaterial zur späteren Präsentation recherchiert sowie mögliche Interaktive Elemente, beispielweise ein Quiz zur Festigung der Lehrinhalte, erstellt.

UbiActive – Die UbiActive-Phase findet während des eigentlichen Besuchs statt. Die im Vorbereitungsschritt erarbeiteten Inhalte werden, abhängig von dem durch GPS bestimmten Standort präsentiert. Weiterhin hat der Anwender die Möglichkeit, eigene Inhalte, seien es

Videoclips, Fotografien oder Notizen hinzuzufügen um später in der Nachbereitung darauf zugreifen zu können.

UbiEducate – Im letzten Schritt werden die in der Aktiv-Phase gesammelten Inhalte aufbereitet und auf einer Zugangsgeschützten Web-Plattform dargestellt. Somit lassen sich gemachte Erkenntnisse und Erlebnisse nachhaltig archivieren.

2.1.2 Komponenten

Für die Durchführung werden folgende Komponenten eingesetzt:

Hardware

Wie bereits beschrieben nutzt das UbiZoo Projekt das Positionsbestimmungssystem GPS. Dadurch ist der Nutzer nicht an einen Beschränkten Vorrat an Positionen gebunden, sondern kann die POIs, vorausgesetzt GPS ist dort nutzbar, frei verteilen. Die technischen Beschränkungen erlauben aber keine Nutzung in geschlossenen Räumen, da das Signal der Satelliten durch Mauerwerk geschwächt beziehungsweise blockiert wird.
Zur Darstellung der Informationen kommen als Clients PDAs und als Informationsserver ein UMPC zum Einsatz, die per Bluetooth Daten austauschen.

Software

Auf der oben genannten Hardware kommen für das Microsoft Dot.Net Framework 3.0 entwickelte Programme zum Einsatz. Für die Darstellung der Inhalte wird Micrsoft Silverlight, Microsofts Adobe-Flash Alternative, genutzt. Für die persistente Speicherung der erstellten Daten wird Microsofts Datenbank Sql Server 2005 eingesetzt.

2.1.3 Kritik

Betrachtet man die Einschränkungen des GPS-Systems hinsichtlich der Planung eines Zoo-Besuchs wird deutlich, dass bestimmte Orte wie zum Beispiel innen liegende Aquarien nicht erreichbar sind. Auch die Winterzeit, in der viele Tiere in Innengehegen gehalten werden, lässt die Einsatzmöglichkeit erheblich schrumpfen.

Weiter lässt sich die Geringe Verbreitung und Vorhandensein der Hardware kritisieren. Im Vergleich mit Mobiltelefonen haben PDAs und vor allem UMPCs eine sehr geringe Verbreitung (Stichprobe von 11511 Haushalten, 4,9% haben ein PDA, 80% ein Mobiltelefon

(Statistisches Bundesamt, 2007a)). Dies bedeutet, dass Nutzung des UbiZoo-Projekts Kauf oder Ausleihe der Geräte erfordert und somit meist weitere Kosten verursacht.

2.1.4 Einordnung

Das UbiZoo-Projekt ist, mit kleinen Einschränkungen, eine mögliche Anwendung der von dieser Arbeit angestrebten Programm-Lösung. Betrachtet man die Art der Dienstbereitstellung ist allerdings ein Unterschied zu erkennen: Das UbiZoo Projekt setzt auf einen mobilen Server, der je nach seiner Position einen speziellen Dienst bereitstellt, wohingegen die Zielsetzung dieser Arbeit auf eine stationäre Infrastruktur abzielt deren Dienstangebot ständig, so lange man sich in Reichweite befindet, vorhanden ist.

2.2 SociaLight

Socialight (Kamida, 2008b) ist ein von der Firma Kamida (Kamida, 2008a) aus New York entwickeltes „Social Network“, das Nutzern erlaubt Daten und Informationen zu Orten hinzufügen und diese anderen Nutzern zur Verfügung zu stellen.

2.2.1 Gliederung

Das Produkt Socialight gliedert sich in zwei Hauptkomponenten, zum einen eine stationäre WWW basierte Infrastruktur und Datenbank, zum zweiten ein auf aktuellen Mobiltelefonen lauffähiges Programm zur Nutzung der Daten, alternativ ein Browser mit dem die WWW-Seite besucht werden kann.

Stationäre Infrastruktur – Die über (Kamida, 2008b) erreichbare Webseite bietet angemeldeten Benutzern die Möglichkeit, auf einer von Google Maps (Google, 2008) bereitgestellten Weltekarte, sogenannte „Sticky Notes“-Markierungen zu verteilen. Diese „Sticky Notes“ können dann mit Informationen zu der gewählten Örtlichkeit versehen und anderen Nutzern zur Verfügung gestellt werden.

Mobilteil – Der Mobilteil bietet die gleichen Administrationsmöglichkeiten wie der Stationärteil. Entweder per Zugriff auf die Socialight-Webseite (*WAP*) über den Browser des Mobiltelefons, oder durch ein Java Mobile Edition (JME) Programm das ebenfalls per Datenverbindung, *GPRS* oder *UMTS*, mit einem Server kommuniziert. Die Java-Anwendung bietet Vorteile hinsichtlich der Lokalisierung des Nutzers, da Sie etwaig vorhandene GPS-Module

Abbildung 2.1: Socialight – Sticky Note

nutzen kann. Da viele aktuelle Mobiltelefone zudem die Möglichkeit bieten, Video- und Audio-Daten aufzunehmen ermöglicht es die Software die „Sticky Notes“ mit diesen Daten zu versehen und zu publizieren.

2.2.2 Ortung

Die Ortung im Socialight-Netzwerk funktioniert auf verschiedene Weise. Wie bereits erwähnt kann durch Nutzung eines GPS-Empfängers die mobile Anwendung die aktuelle Position des Benutzers erkennen und verarbeiten. Sollte das Mobiltelefon keine GPS Funktion unterstützen, gibt es zwei weitere Ortungsvarianten. Zum einen die Positionierung durch Nutzung von vom Mobilfunk-Netzwerk bereitgestellten Daten und zum zweiten die explizite Eingabe der aktuellen Adressinformationen. Für ausgewählte Mobilfunkbetreiber besteht die Möglichkeit durch senden einer SMS an eine bestimmte Nummer (Shortcode), Socialight zu erlauben das Mobiltelefon zu orten. Diese Ortung geschieht anhand der Mobilfunkantennen, bei denen das Telefon aktuell angemeldet ist, dadurch lässt sich eine, je nach Anzahl Antennen in der Umgebung genauere oder gröbere, Position berechnen. Ein Verfahren das heutzutage schon von Rettungsdiensten zur Lokalisierung einer sich in Not befindenden Person genutzt wird.

2.2.3 Datenzugriff

Nachdem der Benutzer geortet wurde, kann er auf die in seiner Umgebung erstellten „Sticky Notes“ zugreifen und anhand deren dann Entscheidungen für seinen weiteren Weg treffen. In der Umgebung befindende „Sticky Notes“ zu erweitern oder neue Daten hinzuzufügen ist nach Ortung, abhängig von den Zugriffsrechten des Benutzers, ebenfalls möglich.

2.2.4 Kritik

Verglichen mit dem UbiZoo Projekt, das einen ähnlich Funktionsumfang bietet, hat Socialight einen Vorteil, es setzt ausschließlich auf Mobiltelefone als Clientgeräte. Dadurch erreicht es eine Steigerung der Anzahl möglicher Anwender und ermöglich somit mehr Menschen den Zugang. Problematisch wiederum sind die entstehenden Kosten. Aktuell sind Datenverbindungen für Handys noch relativ teuer und da möglicherweise vorhandene Audio und Videodaten, auch nach Komprimierung noch relativ hohe Datenmengen darstellen, kann der Kosten- den Nutzfaktur schnell übersteigen.

Als weiterer Kritikpunkt ist auch hier die eingeschränkte Ortung zu nennen. Die Anzahl der Mobilfunkanbieter die eine Ortung unterstützt ist relativ gering, vier sind es für das Vereinigte Königreich. Allen anderen Nutzern bleiben die Möglichkeiten, wenn vorhanden einen GPS-Empfänger zu nutzen, oder ihre Adresse selbst einzugeben. Bei dem Verfahren, seine Adresse selbst mitzuteilen bleibt allerdings offen, ob es nicht einfacher wäre, in Branchensuchmaschinen die Umkreissuche zu nutzen um spezifische Ziele zu finden. Die Verbindungskosten fielen in beiden Fällen an. Dies hätte allerdings den Nachteil das von der Community generierter Inhalt nicht gefunden werden würde, was ja das Hauptangebot von Socialight ist.

2.2.5 Einordnung

Die Grundfunktionalität von Socialight, also Positionen mit Informationen zu versehen und diese anderen bereitzustellen, ist auch mit der von dieser Arbeit angestrebten Anwendung möglich. Voraussetzung dafür, ist allerdings eine Implementierung mit ähnlichen Mitteln. Also der Einsatz von GPS für die Positionierung und einer Kommunikationsmethode die globale Konnektivität bietet. Um Lokale Verbindungen also z.B. per WLAN-Hotspot oder Bluetooth nutzen zu können, müssten die Zugriffspunkte mit einem zentralen Server verbunden sein, der die angeforderten Informationen liefern kann.

3 Grundlagen

Für ein grundlegendes Verständnis des angestrebten Ziels dieser Arbeit werden im Folgenden die zu Grunde liegenden Konzepte und Technologien erörtert.

3.1 Positionsbestimmung

Bevor darauf eingegangen werden kann, wie sich ein Ortsbasierter Dienst definiert müssen zunächst einmal die Begriffe Position (aus dem Lateinischen: *positio*, „das Setzen, Stellen; Lage, Stellung“) und die Bestimmung derer, also die Positionsbestimmung, geklärt werden.

Position ist ein eher abstrakter Begriff, der auf verschiedenste Weise Interpretiert werden kann. Küpper (2005) teilt zunächst einmal in zwei Hauptgruppen ein: Physikalische oder Virtuelle Position.

3.1.1 Virtuelle Position

Die virtuelle Position ist im Gegensatz zur physikalischen irreal. Das bedeutet es handelt sich um Orte, die in einer virtuellen, meist computergenerierten, Parallelwelt existieren. Da in dieser Arbeit reale Positionen bestimmt werden sollen, wird darauf nicht weiter eingegangen.

3.1.2 Physikalische Position

Das Hauptmerkmal der Physikalischen Positionseinteilung ist das Vorhandensein in der realen, physikalischen Welt. Um eine Position genau angeben zu können ist dabei immer ein Bezugssystem notwendig. Alle bekannten Positionseinteilungen sind also immer relativ zu Etwas zu betrachten. Was dieses Etwas ist soll in der folgenden Kategorieneinteilung verdeutlicht werden:

Räumliche Positionen (Spatial Locations)

Diese Art Positionen sind definiert durch Zahlen und Werte, sogenannte Koordinaten. Diese orientieren sich an einem vorher definierten Bezugssystem und dessen Null Punkt (Ursprung), dieser ist meist Ausgangspunkt der Einteilung.

Ein typischer Vertreter ist die geographische Koordinate die eine Position anhand Längen- und Breitengrade angibt (Koordinate der HAW-Hamburg: 53.556142,10.021818).

Beschreibende Positionen (Descriptive Locations)

Im Gegensatz zu Räumlichen Positionen werden Beschreibende Positionen nicht ausschließlich durch Zahlenwerte sondern durch Namen von Orten und/oder bekannten Landmarken definiert. „Jungfernstieg 1, Hamburg, Deutschland“ ist beispielsweise eine beschreibende Position. Kennt man die Position des Landes sowie der Stadt und relativ dazu die jeweilige Straße, findet sich meist ohne Schwierigkeiten auch das durch die Hausnummer markierte Gebäude und somit die gewünschte Position (vorausgesetzt es gibt nur einen Jungfernstieg in Hamburg). Ein dieser Tage weniger verwendeter Einsatz von beschreibenden Positionen ist die Schatzkarte. Auf einer Schatzkarte waren markante Punkte markiert, anhand derer man zu der meist mit „X“ markierten Schatzstätte navigieren konnte.

Netzwerk Position (Network Location)

Diese Art von Positionierungsdaten wird, wie der Name impliziert, für vernetzte Strukturen verwendet. In diesen Strukturen gibt es eindeutig gekennzeichnete Orte, beispielsweise durch eine Folge von Ziffern. Ein Beispiel hierfür wäre das Internet und damit das IP-Protokoll. Anhand einer Ziffernfolge kann somit die zu den Knotenpunkten relative Position bestimmt werden.

3.1.3 Positionsbestimmung

Der Vorgang der Positionsbestimmung dient dazu einer Person oder einem Objekt eine eindeutige Position zuzuordnen, sie oder es zu lokalisieren. Hierbei ist der gewünschte Detailgrad, also Beispielsweise die Auflösung der Koordinaten bei spatialer Positionierung, sowie die damit verbundene benötigte Infrastruktur, vom späteren Verwendungszweck abhängig. Reicht es in einer Büroumgebung aus, den Raum in der sich die zu lokalisierende Person

befindet zu bestimmen um einen ankommenden Anruf an den am geringsten entfernten Apparat zu vermitteln, ist diese Genauigkeit in einem Regalhochlager in dem autonome Roboter Waren transportieren, in keinem Maße ausreichend um etwaige Kollisionen zu vermeiden.

Im Folgenden werden die gängigsten Positionsbestimmungsverfahren zusammengefasst.

Annäherungsmessung (Proximity Sensing)

Bei der Annäherungsmessung bauen eine Anzahl Basisstationen mit Hilfe von in der Reichweite eingeschränkter Messtechnik ein Sensornetz auf. Wenn nun, je nach Art der Nutzung, entweder ein Endgerät das Signal einer Station auffängt, oder die Station das Signal eines Endgerätes registriert, bedeutet dies das Endgerät befindet sich in der Nähe der jeweiligen Basisstation, deren Position bekannt ist. Abhängig von der Anzahl der Stationen und der Reichweitenmodulierung kann somit ein relativ hoch aufgelöstes Positionierungsnetz aufgebaut werden.

Entfernungsmessung (Lateration)

Grundlage der Entfernungsmessung bildet, vergleichbar mit der Annäherungsmessung, ein Sensornetzwerk. Anders als bei der Annäherungsmessung, bei der Kontakt zu einer Basisstation ausreicht, wird bei der Entfernungsmessung der Abstand zu mindestens drei bzw. vier Stationen benötigt, je nach dem ob zwei- oder dreidimensionale Positionierung benötigt wird. Anhand dieser Abstände in Verbindung mit den bekannten Positionen der Basisstationen kann man nun die Position extrapolieren.

Winkelpeilung (Angulation)

Wie bei den zuvor erörterten Methoden wird auch hierbei auf ein Netzwerk von Basisstationen zugegriffen. Weiterhin benötigt, je nach Nutzungsform, entweder das Endgerät oder die Basisstation Ausrüstung zum messen des Einfallswinkels des zu ortenden Signals (Antennenarray). Peilt man an den, entweder an der Basisstation oder am Endgerät, gemessenen Einfallswinkeln entlang, sollte sich ein Schnittpunkt ergeben. Dieser Schnittpunkt ist die Position des zu lokalisierenden Objekts.

Koppelnavigation (Deduced Reckoning, „Dead Reckoning“)

Die Koppelnavigation ist ein aus der Seefahrt stammendes Verfahren, dass anhand der Ausgangsposition „gekoppelt“ mit Geschwindigkeit, Richtung sowie verstrichener Zeit die aktuelle Position vorhersagt. Voraussetzung zum Einsatz dieses Verfahrens ist somit eine bereits bekannte Position und zusätzliche Sensorik die Änderungen des Bewegungszustandes erfasst.

Mustererkennung (Pattern matching)

Grundlegendes Verfahren bei der Mustererkennung ist die Positionsbestimmung anhand von bereits bekannten Daten. Dabei lassen sich zwei Unterkategorien aufstellen. Die Optischen und die Nichtoptischen Musterkennungen.

Bei der Optischen Mustererkennung wird ein per Kamera oder ähnlichen optischen Sensoren aufgezeichnetes Bild verarbeitet. Nach Vorverarbeitung zur Ausdünnung der Bildmerkmale werden diese mit einer Datenbank abgeglichen in der alle bekannten Objekte verzeichnet sind. Bei Vorhandensein eines Merkmals lässt sich dieses dann auf eine Position abbilden.

Die Nichtoptische Mustererkennung basiert auf nichtvisuellen Daten, z.B. Elektrische Feldstärke oder Namen von Umgebenden Drahtlosnetzwerk Zugriffspunkten. Ein Beispiel für die Positionierung anhand von WLAN-Netzen ist Skyhooks Wi-Fi Positioning System (SKYHOOK Wireless, Inc, 2008) das im Apple IPhone (Apple Inc, 2008) zum Einsatz kommt.

3.1.4 Location Services (LCS)

Dienste die die oben genannten Methoden einsetzen um anhand dieser Positionsdaten zu ermitteln heißen Location Services (LCS). Meist besteht ihre Aufgabe darin, Anfragen hinsichtlich des aktuellen Aufenthaltsorts eines Objekts oder einer Person zu beantworten. Dabei beinhalten Anfragen an Location Services laut Küpper (2005) mindestens folgende Daten: das Gebiet in dem positioniert werden soll, die Art des Rückgabewertes und die benötigte Genauigkeit. Die Antwort (Location Data) setzt sich wiederum zusammen aus dem Aufenthaltsort, der Art des bestimmten Ortes (aktueller, letzt bekannter oder Ursprungsaufenthaltsort), dem Format der Positionsdaten, der Qualität beziehungsweise der Auflösung der ermittelten Daten, einer eindeutigen Identifikation sowie der Richtung und Geschwindigkeit des Gesuchten.

3.1.5 Einordnung

Für diese Arbeit von größter Bedeutung sind die Physikalischen Positionen sowie die Bestimmung dieser. Virtuelle Positionen sind für den angestrebten Einsatzzweck nicht von Interesse, da die Dienstleistungen alleine für real existierende Personen angeboten werden sollen.

Die Art Positionsbestimmung sollte, wenn möglich, keinen Einfluss auf Funktionalität der zu entwickelnden Anwendung haben. Um dies zu erreichen bietet sich die Abstraktion per Location Service an. Dieser versteckt die darunterliegende Positionierungs-Infrastruktur und ermöglich eine unabhängige Verwendung der Daten.

3.2 Ubiquitous Computing

Ubiquitous sinngemäß übersetzt bedeutet in etwa Allgegenwärtig. Damit ist allerdings nicht gemeint, dass überall wohin man blickt ein Computer oder ein ähnliches Datenverarbeitendes Gerät seine Arbeit verrichtet. Das Gegenteil ist der Fall.

Mark Weiser, der während seiner Zeit am PARC (Xerox Palo Alto Research Center, 1987-1994 (Mark Weiser, 2008)) den Begriff „Ubiquitous Computing“(UbiComp) geprägt hat, beschreibt die Allgegenwärtigkeit eher als „augmentation“, also als Steigerung oder Vermehrung der „wundervollen Nuancen der realen Welt“. Laut seiner Aussage, kann dabei auf ein explizites personengebundenes Ein-/Ausgabegerät (in seinem Beispiel von 1993, ein PDA) verzichtet werden, da zum UbiComp-Paradigma der unmittelbare und nicht örtlich beschränkte Zugang zu Informationen gehöre.

Streng nach Definition fällt die von dieser Arbeit angestrebte Lösung somit nicht in den Bereich UbiComp, da mobile Endgeräte zur Datenein- und Datenausgabe vorgesehen sind. Diese lassen sich eindeutig in die Kategorie personengebunden einordnen. Wozu also die Erwähnung von UbiComp in diesem Zusammenhang?

Nicht ohne Grund hieß die dieses Jahr zum zehnten Mal stattfindende „UbiComp 2008: Tenth International Conference on Ubiquitous Computing“ früher einmal „HUC – International Symposium on Handheld and Ubiquitous Computing“ (UbiComp, 2008). Zwar kann man heutzutage mehr denn je dem Anspruch nach unmittelbarer Verfügbarkeit von Informationen gerecht werden, allerdings fristen etwaige Produkte immer noch ein Nischendasein und sind daher kaum verbreitet (zum Beispiel „Wearable Computer“ z.B. (Apple, 2007)).

Die ersatzweise Nutzung von mobilen Endgeräten kann, durch die bereits hohe Verbreitung, somit als ein Weg betrachtet werden, Ubiquitous Computing zu mehr Popularität zu verhelfen. Jene Art Popularität, die auch schon im Falle des World Wide Webs zu einer Art Hype

geführt und die Nachfrage exponentiell gesteigert hat. Je mehr die Popularität von UbiComp wächst, desto mehr Produkte und Geräte die sich für das in dieser Arbeit angestrebte Ziel eignen werden in den Markt Einzug halten.

3.3 Mobile Endgeräte

Das folgende Kapitel gibt einen Überblick über Eigenschaften, Einsatzgebiete und die Entwicklung von mobilen Endgeräten (ME). Ziel ist es, die Abgrenzung von nicht mobilen Geräten zu verdeutlichen und Einschränkungen hinsichtlich der Nutzbarkeit aufzuzeigen, da diese Beschränkungen starken Einfluss auf das Design der zu entwickelnden Anwendung haben können.

3.3.1 Definition

Das globale voranschreiten der Forschung auf dem Gebiet der Werkstofftechnik und die daraus resultierenden Möglichkeiten der Miniaturisierung von Elektronikteilen, ermöglichen uns heutzutage die Herstellung von Komponenten, die sowohl mit ihrer Leistungsfähigkeit als auch mit ihren geringen Abmessungen überzeugen. Mobile Endgeräte sind das Resultat dieser Entwicklung.

Als Endgerät bezeichnet man ein Gerät, das ein Endpunkt eines Datennetzes ist. Mobil bedeutet im Allgemeinen tragbar oder beweglich. Daraus lassen sich also zwei Anforderungen ableiten die ein Gerät zu einem Mobilen Endgerät machen:

1. Das Gerät muss teil eines Netzwerkes sein, also die Möglichkeit der Kommunikation mit anderen Geräten bieten.
2. Das Gerät darf den Nutzer nicht in seiner Beweglichkeit einschränken, sei es durch zu hohes Gewicht oder aber durch Kabelgebundene Energieversorgung die nur einen gewissen Radius um eine Energiequelle zulässt.

3.3.2 Eigenschaften

Entscheidend für die Nutzung eines mobilen Endgerätes und für die Unterscheidung zu Stationärgeräten sind dessen Eigenschaften.

Abbildung 3.1: Mobile Endgeräte

Energiezufuhr / Leistung

Um eine relativ hohe Mobilität erreichen zu können, benötigt das ME eine eingebettete Energiequelle. Heutzutage sind das meist Akkus oder vereinzelt Brennstoffzellen. Obwohl die Bestandteile der Geräte immer weiter hinsichtlich ihres Stromverbrauchs optimiert werden und auch die Speicherkapazitäten von Batterien stetig zunehmen, bleibt die Energieknappheit der maximal beschränkende Faktor. Entwickler von MEs müssen daher abwägen, ob mehr Laufzeit oder z.B. ein größerer Anzeigebereich wichtiger ist.

Energie- und Platzknappheit begrenzen natürlich die Leistungsfähigkeit der Geräte. Virtueller- bzw. Massenspeicher, Grafikbeschleunigung, Rechengeschwindigkeit und Multimediafunktionalität sind daher nur selten mit zu immobilen Geräten vergleichbarer Kapazität vorhanden.

Erweiterbarkeit

Eine weitere Eigenschaft von MEs ist die erschwerte hardwareseitige Erweiterbarkeit. Zwar bringen viele MEs Schnittstellen mit, diese sind aber meist proprietär und nur in geringer Stückzahl vorhanden. So dass eine nachhaltige Leistungssteigerung über Zusatzkomponenten erschwert wird. Die Hauptbestandteile sind hoch integriert und verglichen mit Standard Desktop-PC-Teilen nur in geringem Maße zugänglich beziehungsweise austauschbar, wobei der Mangel an passenden die Leistung steigernden Teilen ein Übriges tut.

Betriebssystem

Ähnlich dem normalen Personal Computer besitzen auch MEs ein Betriebssystem zur Vermittlung zwischen Hard- und Software. Bekannte Vertreter sind „Symbian OS“ (Symbian Limited, 2008), „Windows Mobile / Windows Compact Edition (CE)“ (Microsoft, 2008b), „Palm

OS“ (Palm Inc., 2008) oder auch verschiedene Linux Derivate. Abhängig von diesem Betriebssystem können Programme entwickelt werden die auf den mobilen Endgeräten lauffähig sind. Da allerdings die Vielfalt von unterschiedlicher Hardware immens groß ist, und auch die Zahl einsetzbarer Betriebssysteme einen nicht zu vernachlässigenden Faktor darstellt, wurden verschiedene mobile virtuelle Laufzeitumgebungen entwickelt, die von Hardware- und Betriebssystemeigenheiten abstrahieren und Programme somit variabel Einsetzbar machen, siehe Kapitel 3.4.

Kommunikation

Um der Definition für Endgeräte gerecht zu werden, wird Zugang zu einem Datennetzwerk benötigt. Da dieser nicht durch eine Kabelverbindung erfolgen darf, dies widerspricht der Mobilitätsanforderung, muss die Kommunikation drahtlos von statten gehen. Am meisten verbreitet sind hierbei folgende Technologien: *GSM/UMTS*, *WLAN*, Bluetooth und *IRDA* (Infrarot). Diese unterscheiden sich vor allem in Übertragungsgeschwindigkeit und Reichweite. In der untenstehenden Tabelle finden sich die gängigen Leistungsmerkmale der genannten Technologien.

Datum/Technologie	**GSM/UMTS**	**WLAN**	**Bluetooth**	**IRDA**
Geschwindigkeit(Mbit/s)	0,015	2 bis 300	2,1 (EDR)	16 (VFIR)
Maximale Reichweite(m)	35.000	Bis 100	circa 100	circa1

Tabelle 3.1: Merkmale von Kommunikationstechnologien

Die Maximale Reichweite bezieht sich hierbei auf hindernisfreie Strecken. Mit einer Isotropen-Antenne und der maximal in Deutschland zulässigen Sendeleistung.

Interaktion

Einen besonderen Stellenwert nimmt die Fähigkeit zur Interaktion mit dem Benutzer ein. Das Vermögen Informationen darzustellen und die Möglichkeiten des Nutzers mit dem Gerät zu interagieren sind hierbei von besonderer Wichtigkeit. Bei den aktuell verbreiteten Endgeräten variieren diese Eigenheiten stark. Mobile Endgeräte besitzen, wie eigentlich alle eingebetteten Systeme, einen Hauptfunktionszweck. Auf diesen hin werden die Interaktionsmöglichkeiten abgestimmt. Als Beispiel sei hier ein Pager genannt. Ein Pager besitzt meist nur Monochromes Display mit geringer Zeilenanzahl und wenigen Bedienelementen. Dies ist der Tatsache geschuldet, dass er nur dem Zweck Nachrichten zu empfangen dient um dann

anderweitig reagieren zu können. Im Gegensatz dazu besitzt eine mobile Spielkonsole eine Vielzahl Eingabeelemente und einen durchaus hochwertigen Anzeigebereich, der sowohl bewegte als auch statische Bilder ansprechend darzustellen vermag. Dieser Vergleich verdeutlicht die Vielfalt unterschiedlicher Eigenschaften, die im Bereich Interaktion mit Mobilen Endgeräten anzutreffen sind.

Sicherheit

Nicht unerwähnt bleiben darf das Thema der Datensicherheit. Wie bereits beschrieben, bedienen sich Mobile Endgeräte drahtloser Kommunikation um ihrer Definition als mobil gerecht zu werden. Immer wenn Information über das Medium Luft übertragen werden besteht die Möglichkeit, dass die Daten abgefangen, aufgezeichnet oder verändert werden. Die beste Methode, um dies zu verhindern beziehungsweise die Daten für den Angreifer unbrauchbar zu machen ist es eine Verschlüsselung einzusetzen. Aufgrund mangelnder Leistungsfähigkeit ist dies allerdings nicht immer Möglich, da Verschlüsselung mit ausreichend langen Schlüsseln kurzzeitig hohe Anforderungen an das System stellt und damit die Reaktionszeiten des Gerätes negativ beeinflussen kann. Die meisten hier genannten Übertragungstechnologien unterstützen daher Hardwareseitige Verschlüsselung um den Prozessor zu entlasten.

3.3.3 Einordnung

Der Begriff mobiles Endgerät schließt eine Vielzahl verschiedenster Geräte ein. Deren Eignung für ein Projekt wie dieses hängt in besonderem Maße davon ab, wie gut Informationen vermittelt werden können und welchen Leistungsumfang das Gerät bei der Benutzereingabe bietet. Auf diese beiden Merkmale muss daher ein besonderes Augenmerk gerichtet werden. Zusätzlich ausschlaggebend für die Wahl ist die Verbreitung der Geräte. Je verbreiteter ein Gerät ist, umso akzeptierter ist es und desto besser eignet es sich für die angestrebte Anwendung.

3.4 Mobile Laufzeitumgebungen

„Wozu dient eine mobile Laufzeitumgebung?“, „Wie unterscheidet sie sich von ihrem immobilen Pendant?“, „Welche Vor- und Nachteile bietet Sie?“, „Welches sind die Bekanntesten und am meisten Verbreiteten mobilen Laufzeitumgebungen?“, diese Fragen klärt der nächste Abschnitt.

3.4.1 Definition

Mobile Laufzeitumgebungen schaffen eine Ebene zwischen Hardware/Betriebsystem und der ausführbaren Software. Genauer bedeutet dies, die Software nutzt Komponenten die ihr von der Laufzeitumgebung (Virtuelle Maschine) zur Verfügung gestellt werden, diese übersetzt dann die Anforderungen in die jeweilige Hardware/Betriebssystem-Domäne. Dies verhindert die Notwendigkeit des direkten Zugriffs auf die jeweiligen Ressourcen (Enge Kopplung) und somit die nötige Anpassung an sämtliche Hardwarekonfigurationen.

Dieses Konzept ermöglicht dem Anwendungsentwickler ein Maximum an Programmportabilität. Relativiert wird der Grad der Portabilität allerdings noch von der Verbreitung der Laufzeitumgebung. Im Vergleich zu Laufzeitumgebungen für immobile Endgeräte müssen die mobilen Vertreter mit stark eingeschränkten Ressourcen arbeiten. Dabei ist besonders das geringe Speicherangebot, flüchtig und nichtflüchtig, von Bedeutung. Kompensierung dieser Einschränkung wird meist durch Verkleinerung des Methodenbestandes und/oder Objektbestandes der Basisbibliotheken erreicht. Daraus folgt, dass mobile Laufzeitumgebungen zwecks optimaler Lauffähigkeit häufig auf Funktionalität verzichten.

3.4.2 Einordnung

Bereits in der Zielsetzung (Abschnitt 1.2) ist von der Maximierung von zu unterstützenden Clientgeräten die Rede. Daher sind für dieses Projekt besonders die von einer Laufzeitumgebung gebotene Abstraktion vom Betriebssystem und die damit verbundene hohe Portabilität interessant. Durch den Einsatz einer Laufzeitumgebung kann das zu entwickelnde Programm auf allen Geräten Verwendung finden, für die eine Implementierung der jeweiligen Virtuellen Maschine existiert.

3.5 Ortsbasierte Dienste

Das Angebot von ortsbasierten Diensten ist eine Hauptaufgabe der zu konzipierenden Anwendung. Der nächste Abschnitt beschreibt was genau unter ortsbasierten Dienstleistungen zu verstehen ist.

3.5.1 Definition / Anwendung

„Location Based Services“ (LBS) oder übersetzt ortsbasierte/ortsbezogene Dienste sind Dienste, die dem Nutzer zu seiner aktuellen Position kontextbezogene Funktionalität anbie-

ten. Ein häufig bemühtes Beispiel ist es, dem Benutzer anhand seiner Position und seiner spezifizierten Vorlieben ein geeignetes geographisch nahes Restaurant zu empfehlen.

Eine Definition vom Standard für Mobilfunknetze GSM (Global System for Mobile Communications, (GSM Association, 2008)) vom LBS lautet wie folgt:

„Location Based Services (LBSs) are services that use the location of the Target for adding value to the service... As the term LBS covers the service aspects it should not be mixed up with the term LoCation Services (LCS),..., which covers all the hardware e.g. network elements and entities necessary to provide LBS..." (Association, 2003)

Zusätzlich zu dem bereits benannten Wertgewinn ist hier von der Notwendigkeit eines Location Service (LCS, vgl. Abschnitt 3.1) die Rede. Ohne die Positions-Information die der LCS zur Verfügung stellt, könnte ein LBS keinerlei Kontextdaten bereitstellen, da der Kontext nicht ermittelbar wäre.

Da Positionsdaten teilweise in Koordinaten ermittelt werden, mit denen ein Anwender ohne Zugriff auf ein GIS nur wenig anfangen kann, ist eine Hauptaufgabe von LBS auch die Konvertierung von verschiedenen Positionsdatenformen. Als typische Anwendung sei hier die Bestimmung der nächsten Filiale einer Geschäftskette genannt: Abfrage der spatialen Position (Koordinaten) anhand der Deskriptivposition (Adresse des Abfragenden), Berechnung der Entfernungen zu den Filialen, Ausgabe der Deskriptivposition der geographisch am geringsten entfernten Filiale.

3.5.2 Bestimmung

LBS gehören zu den sogenannten Context Aware Services (CAS), also den kontextbewussten Diensten. CAS sind im Vergleich zu LBS generischer gefasst. Der Kontext auf den sie sich beziehen ist nicht wie bei den LBSs ein rein geographischer Kontext, sondern kann sich auch auf andere Grundparameter beziehen. Laut Schmidt u. a. (2003) wären dies zum Beispiel die „Human Factors" also Personenbezogene Parameter: Wie fühlt sich die Person (User) oder welche Aufgabe hat sie gerade zu erfüllen (Task).

CAS beziehen ihre Informationen aus der Auswertung von Sensorenwerten. Dabei stellen die Rohdaten den „Primären Kontext" dar. Durch Kombination, Filterung und Verfeinerung der Primärdaten wird der „Sekundäre Kontext" erzeugt. Dieser stellt dient dann als Informationsbasis des CAS.

Ein LBS Beispiel wäre also: Kombination aus GPS- und Automobiltelemetrie-Daten(Geschwindigkeit, Richtung, Primärer Kontext), um in einem Tunnel die aktuelle Position zu bestimmen (Sekundärer Kontext, Kombination der Daten „Dead Reckoning")

und anhand derer die aktuell geltende Geschwindigkeitsbegrenzung anzuzeigen („Added value“).

3.5.3 Kategorisierung

Küpper (2005) unterteilt Location Based Services zwecks deutlicherer Unterscheidung hinsichtlich ihrer Nutzung in zwei Kategorien: Reaktiv und Proaktiv. Im Fachgebiet Informatik kann man diese wohl am Besten mit „Push“(Proaktiv) und „Pull“(Reaktiv) gleichsetzen.

Proaktiv

Proaktive Dienste benötigen, wenn überhaupt, nur eine einmalige Aktivierung. Danach wird die Nutzerposition ständig überwacht („Tracking“) und ausgewertet. Zum Einsatz kommen könnte so ein proaktiver Dienst bei Sportveranstaltungen, wie einem Handballspiel. Die Spielerpositionen sowie die Ballposition würden ständig überwacht werden. Sobald der Ball die Torlinie überschreitet könnte der Ursprungsort berechnet und dargestellt werden.

Reaktiv

Im Gegensatz zu proaktiven Diensten erfordern reaktive Dienste jedes Mal eine explizite Aktivierung durch den Nutzer. Die oben erwähnte Anwendung der Lokalisierung von POIs ist ein Beispiel für einen reaktiven Dienst. Die Dienstleistung wird nur explizit auf Anfrage hin ausgeführt, die Nutzerposition wird somit nur nach Aktivierung zwecks Entfernungsmessung ermittelt.

3.5.4 LBS Supply Chain

Küpper (2005) beschreibt unter dem Überbegriff „Supply Chain“, also Versorgungskette, verschiedene Rollen und Aktoren die bei der Nutzung eines Ortsbasierten Dienstes in Erscheinung treten können. Ein Aktor wird definiert als Individuum oder Organisation die anderen Aktoren Dienste anbietet, deren Dienste nutzt oder beides. Eine Rolle ist beschrieben als ein Tätigkeitsfeld eines Aktors. Zudem gibt es noch die sogenannten Referenzpunkte, diese stehen stellvertretend für die Interaktion zwischen verschiedenen Rollen.

Die typischen Bestandteile eines LBS-Systems und ihre Zuordnung lassen sich der Abbildung 3.2 entnehmen.

Target

Ein mobiles Objekt oder eine mobile Person die oder dessen Position bestimmt werden soll.

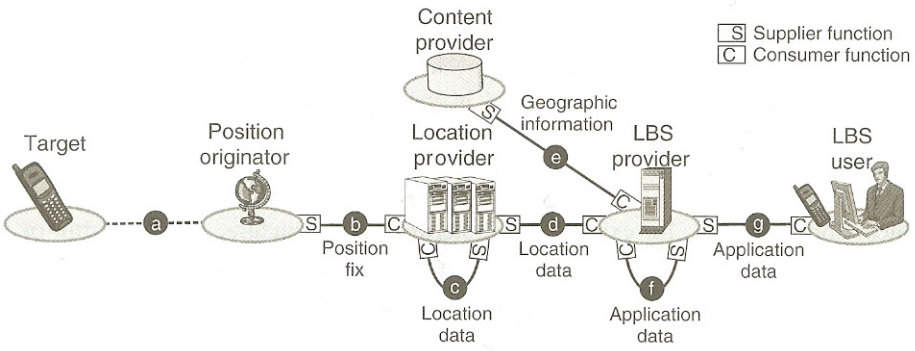


Abbildung 3.2: LBS Supply Chain aus Kuepper:2005

Position originator

Berechnet die Position (Primärer Kontext).

Location provider

Verarbeitet die Position und stellt die erzeugte „Location Data“ bereit (Sekundärer Kontext).

LBS provider

Der Systembestandteil, der die Informationen, also den Wert, zusammenträgt und bereitstellt.

Content provider

Stellt dem LBS provider Inhalt zu den überwachten Orten zur Verfügung.

LBS user

Anwender der sich über seinen Standort oder den Standort eines Targets Informationen beschaffen möchte.

3.5.5 Einordnung

Der angestrebte Serverteil entspricht der Definition eines Location Bases Services. Er fügt der Umgebung "Wert“ hinzu, indem er diesbezüglich Informationen und Dienste anbietet. Die zu entwickelnde Client-Anwendung und deren Benutzer sind somit gleichzusetzen mit dem Target beziehungsweise dem LBS user. Die Art des Dienstes, ob pro- oder reaktiv, ist abhängig von der jeweiligen Positionierungsinfrastruktur und der Art der Anwendung. Am Sinnvollsten erscheint die Nutzung einer Kombination der beiden Verfahren. Proaktive Verarbeitung der Positionsdaten um dem Anwender etwaig vorhandene Dienste zu präsentieren. Sowie

reaktive Nutzung dieser Dienstleistungen (Anwender fordert explizit Informationen an). Eine proaktive Informationsübermittelung ist nur in ausgesuchten Fällen erstrebenswert, übermäßige Nutzung könnte schnell als *Spam* betrachtet werden.

3.6 Bluetooth

Der folgende Abschnitt beschreibt die Grundlagen der Bluetooth-Kommunikationstechnologie. Er dient dazu, zu verdeutlichen in wie und in welcher Form sie sich für den Einsatz in diesem Projekt eignet.

3.6.1 Definition

Bluetooth ist eine im lizenzfreien ISM-Band (2,4 Ghz) arbeitende drahtlose Kurzstrecken-Übertragungstechnologie bei deren Entwicklung besonderes Augenmerk auf Energieeffizienz gelegt wurde. Ihre ursprüngliche Aufgabe war die der Vermeidung von Kabeleinsatz bei peripherem Mobilfunkzubehör. Heutzutage umfasst die Palette der Einsatzgebiete eine Vielzahl von verschiedensten Lösungen. Drahtlose Eingabegeräte (Tastatur, Maus und Joystick), Ausgabegeräte (Kopfhöhrer, Headsets oder digitale Bilderrahmen), Datengeräte (GPS-Empfänger und Webcam) sind dabei nur ein kleiner Auszug. Der Technologiestandard mit dem etwas befremdlichen Namen, wurde 1994 von der Firma Ericsson (Ericsson, 2008) vorgestellt. Um einen global erfolgreichen Standard schaffen zu können, setzten die Entwickler auf einen offenen Standard und nicht auf einen Proprietären. Das ermöglichte weiteren Großunternehmen die Technologie einzusetzen und bei der weiterentwicklung zu partizipieren. 1998 gründeten Intel, IBM, Nokia und Toshiba zusammen mit Ericsson die sogenannte Bluetooth Special Interest Group (Bluetooth SIG). Der Hauptweck dieses stetig wachsenden Zusammenschlusses von Großunternehmen ist die Weiterentwicklung und Fortführung des Bluetooth Standards. Er umfasst heute mehr als 10000 Unternehmen (Stand Juli 2008).

3.6.2 Bluetooth-Stack

Der Bluetoothstack definiert die verschiedenen Schichten und Protokolle die bei einer Bluetoothverbindung eingesetzt werden können. Er setzt sich aus zwei Komponenten zusammen, der oberere Bereich wird als „Bluetooth host“-Komponente bezeichnet. Er wird meist in Software umgesetzt, als Bestandteil des jeweiligen Betriebssystems. Der untere Bereich ist der sogenannte „Bluetooth controller“ oder auch „Bluetooth radio module“ – Bluetooth Funkmodul. Dieser ist normalerweise eine Hardwarekomponente die auf unterschiedlichste Art mit dem Host-System, also dem System auf dem der Bluetooth-Stack läuft, interagiert.

Bluetooth Protokolle

Ein Protokoll beschreibt den Ablauf und die Struktur beziehungsweise das Format einer Kommunikation. Im Bluetoothstandard sind verschiedene Grundprotokolle definiert, mit deren Hilfe sich verschiedenartige Anwendungen sowie im Stack weiter oben angesiedelte Protokolle realisieren lassen. Die Kernprotokolle, also die Basis, bilden das Basisband-, das Link Management- und das Service Discovery Protocol (SDP). Aufgabe des Basisbands und des Link Managements ist die Steuerung der Verbindungen. Mit Hilfe des Service Discovery Protocol können Geräte ihre Umgebung und etwaig vorhandene Bluetoothgeräte nach zur Verfügung gestellten Diensten durchsuchen oder Geräteinformationen abfragen.

Ein im Zusammenhang mit dieser Arbeit interessantes nicht Kernprotokoll, ist das RFCOMM-Protokoll (Radio Frequency Communication). Es emuliert eine serielle Verbindung. Daten lassen sich nach Aufbau der Verbindung in beide Richtungen übertragen.

Bluetooth Profil

Die Bluetooth SIG hat verschiedene sogenannte „usage models“ oder auch Einsatzmodelle festgelegt. Diese unterscheiden sich in ihrer Grundfunktionalität und ihrem Einsatzzweck. Diese Einsatzmodelle werden durch Bluetooth Profile repräsentiert. Ein Profil definiert verschieden Eigenschaften beziehungsweise Protokolle die für die Erfüllung erforderlich sind. Das allgemeinste und wichtigste Protokoll, dass von allen Bluetoothgeräten unterstützt werden muss, ist das GAP (Generic Access Profile). Es definiert Routinen und Prozeduren zur Steuerung der Verbindungen. Die für diese Arbeit wichtigen Profile werden im Folgenden zusammengefasst.

SDAP (Service Discovery Application Profile) – SDAP enthält eine Anwendung die mit Hilfe des Service Disvcovery Protocols (SDP) feststellen kann welches Gerät welche Dienste anbietet.

SPP (Serial Port Profile) – Das Serial Port Profile dienst der Verbindung zweier Bluetooth-Geräte per serieller Verbindung. Es wurde entwickelt, um serielle Kabelgebundene Geräte möglichst einfach über Bluetooth nutzen zu können. Basis für das SPP bilde das RFCOMM-Protokoll.

3.6.3 Einordnung

Die hohe Verbreitung von Bluetooth in Mobiletelefonen (Group, 2007), bedeutet auch das durch eine Nutzung in diesem Projekt, eine große Zielgruppe erschlossen werden kann.

Zusätzlich ist die durch Nutzung des ISM-Bandes kostenlose Übertragung von Daten einer Akzeptanz durch den Endverbraucher sehr zuträglich. Zu beachten sind aber auch die Einschränkungen, die sich durch die ursprüngliche Ausrichtung als Kabelersatzprotokoll ergeben, zum Beispiel die niedrige maximale Teilnehmerzahl von 8 Geräten oder der aktuell noch geringe Datendurchsatz (Der Bluetooth 3.0 Standard sieht eine Kopplung mit *UWB* vor und ermöglicht Datenraten von über 485Mb/s).

3.7 Mobile Dokumententypen

Um Informationen für mobile Endgeräte bereitzustellen, gibt es aktuell zwei Ansätze. Da die Darstellung und die Erstellung von Informationen ein Hauptanliegen dieser Arbeit ist, sollen diese hier kurz vorgestellt werden.

3.7.1 XHTML Mobile Profile

Wie im Abschnitt Anwendung erwähnt existiert eine Untermenge von XML für die Gestaltung für Webseiten. Diese Untermenge beinhaltet die von HTML spezifizierten Elemente. Um auf die fortschreitende Verbreitung von Webfähigen Geräten mit eingeschränkten Ressourcen zu reagieren wurde 2001 von der Open Mobile Alliance (Open Mobile Alliance, 2008) das sogenannte XHTML Mobile Profile entworfen. Dieses Profil ist eine Untermenge von XHTML und eine Übermenge von XHTML Basic. Es dient dazu, Webautoren eine Möglichkeit zu geben, Informationen trotz eingeschränkter Fähigkeiten der Geräte angemessen darzustellen. Die unterstützen Elemente lassen sich der Spezifikation (Wireless Application Protocol Forum, Ltd., 2001) entnehmen.

Einordnung

Das XHTML Mobile Profile stellt eine Möglichkeit da, Informationen auf mobilen Endgeräten darzustellen. Besonders die Kompatibilität zu XHTML macht es interessant, da dieser weitverbreitete Standard unter Webautoren schon recht akzeptiert ist. Die Nutzung dieses Profils würde es Autoren erlauben ihre XHTML Kenntnisse zur Generierung von Inhalten für die angestrebte Anwendung zu Nutzen.

3.7.2 Wireless Markup Language (WML)

Die Wireless Markup Language ist eine auf XML basierende Sprache für die Generierung von Web-Inhalten für mobile Endgeräte. Ebenso wie das XHTML Mobile Profile wurde sie von der Open Mobile Alliance entwickelt. Ihr Ursprung liegt allerdings noch etwas früher als der des Mobile Profiles, um genau zu sein ungefähr drei Jahre (1998). Sie war zu dieser Zeit die meist verwendete Sprache auf diesem Gebiet.

In der aktuellen Version der Spezifikation wird darauf hingewiesen, dass WML nicht mehr für die Gestaltung von Webseiten einzusetzen ist. Stattdessen, solle das oben genannte XHTML Mobile Profile Verwendung finden. Diese neue Spezifikation diene nur der Angleichung der beiden Dokumententypen dahingehend eine Abwärtskompatibilität zu erreichen. Da WML mehrere Konzepte beziehungsweise Elemente und Attribute spezifiziert, die im Mobile Profile nicht vorhanden sind, mussten diese hinzugefügt werden. Ein interessantes Konzept von WML ist der Einsatz von sogenannten Karten und Stapeln (cards and decks). Eine WML-Seite besteht aus einem Stapel von Karten (deck). Jede dieser Karten enthält strukturierten Inhalt und Navigationsinformationen. Ein Nutzer hat die Möglichkeit zwischen diesen Karten hin und her zu springen. Im Vergleich zu einer normalen HTML Seite würde dies bedeuten, dass in einem Dokument mehrere Seiten vorhanden sind. Diese Art der Präsentation lässt sich mit dem geringen Darstellungsgröße erklären. Auf einem mobilen Endgerät mit einem sehr kleinen Display lassen sich nur wenige Informationen darstellen. Um einen Overhead an Verbindungsauf- und -abbau zu vermeiden bietet es sich an mehrere Inhalte in einem Datenstrom zu übertragen und die Navigation dem Endgerät zu überlassen.

Einordnung

Da die WML spezifizierende Open Mobile Alliance (Open Mobile Alliance, 2008) selbst bereits davon abrät, WML für die Inhaltsgenerierung einzusetzen, ist von der Nutzung in diesem Zusammenhang ebenfalls abzusehen.

4 Analyse

Der Abschnitt Analyse befasst sich mit der Bereitstellung der für den Designprozess benötigten Daten. Die bisher getroffenen Aussagen werden hinsichtlich nützlicher Informationen analysiert und mit weiteren, aus Anwendungsfällen gewonnenen, Daten kombiniert.

4.1 UseCases

Use Cases oder auch Anwendungsfälle sind in der Informatik essentielle Bestandteile des Designprozesses. Sie ermöglichen dem Softwarearchitekten und dem Kunden bestimmte Grundfunktionalität der zu erstellenden Anwendung visuell darzustellen. Im Designprozess lassen sich von ihnen auf zu erfüllende Anforderungen schließen. Bei der späteren Realisierung können als Verifikation für den Funktionsumfang herangezogen werden.

Die nun folgenden Anwendungsfälle beschreiben drei verschiedene Szenarien. Aus diesen Szenarien lassen sich zu erstellenden Funktionen und Einsatzmöglichkeiten der konzipierten Anwendung herleiten.

4.1.1 UseCase „Einkaufszentrum“

Ein Anwender (*A*) betritt ein Einkaufszentrum das mit der LBS-Infrastruktur ausgestattet wurde. *A*hat sich zuvor auf dem Webauftritt des Einkaufszentrums über die vorhandenen Geschäfte informiert und daraufhin, unter Angabe von persönlichen Daten, die Clientsoftware auf seinem Mobiltelefon installiert und konfiguriert.

Bei betreten des Kaufhauses wird *A* darauf hingewiesen die Clientanwendung zu starten um etwaig vorhandene Dienste nutzen zu können. Dies geschieht durch visuelle oder akustische Signalisierung. *A* betritt den Einzugsbereich eines Servers der zu einem Musikgeschäft gehört. A wird deshalb, je nach Einstellung der Clientanwendung**,** nun über akustische oder taktile (Vibration) Signale das Betreten des Bereiches mitgeteilt. Der Server bietet *A* die Möglichkeit der Teilnahme an einem Gewinnspiel oder auch verschiedene Musikstücke als Stream zum Probehören an, die A jedoch ignoriert. A setzt seinen Besuch fort und gerät nun in die Nähe eines Schnellrestaurants. Nachdem der Verbindungsaufbau initiiert wurde, wird

A als Bonus für die Nutzung der Software ein digitaler Gutschein angeboten. Dieser kann nur bei der Bestellung über den LBS-Client eingelöst werden. Sobald die bestellten Speisen vorbereitet sind, bekommt A eine Nachricht und kann diese abholen und bezahlen. Alle LBS-Server sind über ein Netzwerk mit einem zentralen Serverrechner verbunden auf dem alle Transaktionen verwaltet werden können um weitere Aktionen, zum Beispiel versenden von personifizierter Werbung oder Bonus/Rabatten, zu veranlassen.

4.1.2 UseCase „Messe“

Anwender (A) betritt ein Messegelände auf dem die Infrastruktur der konzipierten Anwendung vorhanden ist. Am Eingang erfährt er z.B. durch Messe-Hostessen von der Möglichkeit während seines Besuchs Informationen zu sammeln und diese später gebündelt zu erhalten. A willigt ein und bekommt auf seinem Mobiltelefon die Software installiert und wird gebeten diese zu starten. In die Software wurde vorher eine Liste von auf der Messe vorhandenen Kategorien von Ausstellern geladen. A hat somit die Möglichkeit noch bevor er seinen Besuch beginnt auszuwählen welche Kategorien ihn interessieren. Nach der Konfiguration beginnt A sich durch das Gelände zu bewegen. Sobald er in den Einzugsbereich eines Standes gerät, den A als interessant gekennzeichnet hat meldet sich die Software und bietet A an Informationen abzurufen. Bestätigt A dieses bekommt er ausgewählten Text der Ihn über den aktuellen Stand informiert angezeigt. Die Messe-Infrastruktur wird über diese Interaktion informiert, und speichert diese im für A angelegten Profil. Nach Beendigung seines Rundgangs erhält A am Ausgang einen Datenträger überreicht auf dem sich zusätzliche Daten zu den von getätigten Interaktionen befinden. A bekommt somit die Möglichkeit seinen Messebesuch zu Hause nachzubereiten und weitere Nachforschungen anzustellen oder Kontakt aufzunehmen.

4.1.3 UseCase „Tag der offenen Tür“

Zu dem Eingangs erwähnten Szenario des Tages der offenen Tür werden hier noch einige ausgesuchte mögliche Anwendungen dargestellt.

Alle Projekte

- Teilnahme an Studien bezüglich des Gefallens/Nicht Gefallens der einzelnen Projekte.
- Hinterlassen von Nachrichten für andere Benutzer, sogenannte „virtual tags“ (ähnlich den Graffiti-Tags an Wänden).

GameLab:

- Einflussnahme auf ein ausgestelltes Interaktives Spiel, z.B. kreieren von zusätzlichen Gegnern
- Nutzung des Mobilgerätes als interaktives Eingabegerät wenn Lagesensoren vorhanden sind. Kippen oder Beschleunigung führt zu Aktion in einem Spiel.
- Darstellen eines Farbmusters auf dem Display des Mobilgerätes, dieses wird von einer Kamera erfasst und in Positions- und Eingabeinformationen umgesetzt (analog zur Erfassung der C'T-Bots im Projekt Roboterfußball).
- Senden von einem mit der integrierten Kamera erfassten Portrait als Repräsentation eines virtuellen Avatars.

4.2 Anforderungsanalyse

Die Anforderungsanalyse oder auch „Requirement Management" genannt bildet den Grundstock für die zu entwickelnde Anwendung. Das Ergebnis dieses Prozesses ist ein Artefakt beziehungsweise ein Dokument, das für den Entwickler als eine Art Leitfaden für den Entwicklungszyklus darstellt.

Im Folgenden werden die aus den Anwendungsfällen und der Zielsetzung abgeleiteten Anforderungen gelistet und nach ihrer Art, funktional bzw. nicht funktional, sowie ihrer Wichtigkeit für das Endprodukt gegliedert.

4.2.1 Gliederung

Die hier vorgestellten Anforderungen werden in funktionale und nichtfunktionale Anforderungen gegliedert. Zusätzlich erfolgt eine Gewichtung in die bekannten Kategorien: Must, Should und Nice-to-Have. Die Einteilung in diese Kategorien dient der Priorisierung und legt somit die Reihenfolge der späteren Implementierung fest.

4.2.2 Funktionale Anforderungen

(C) steht für Anforderung an die Client-Anwendung.

(S) steht für Anforderung an die Server-Anwendung.

F1 (C) MUST Sinnvoll strukturierte Informationsdarstellung

Ein Hauptanliegen der Anwendung ist die Bereitstellung von Informationen für den jeweiligen Ortskontext. Somit ist die Darstellung dieser eine essentielle Anforderung.

F2 (C,S) MUST Ermittelung von Standortinformationen

Zur Ermittlung der zu der aktuellen Position zugehörigen Information muss diese bekannt sein. Der Client muss auffindbar sein, der Server muss den Client finden können.

F3 (C,S) MUST Reaktion auf Eingabe des Anwenders

Eine Hauptanforderung an ein Interaktives System ist die Reaktion auf Benutzereingaben. Dies bezieht sich auf die Client-Funktionalität. Der Server ist hier nur insofern einzubeziehen, wie er die entstehenden Anfragen beantworten muss.

F4 (C,S) MUST Informationshoheit liegt beim Anwender

Die Anwendung soll den Anwender entscheiden lassen, welche Informationen er beziehen möchte. Forciertes darstellen von Informationen widerspricht einer guten Useability und sollte somit vermieden werden.

F5 (S) MUST Variabler Datenbestand

Informationen besitzen meist eine hohe Änderungsfrequenz. Sie müssen somit leicht austauschbar bzw. aktualisierbar sein. Das bedeutet zum Beispiel nur für die Änderung von Informationen darf keine Neukompilierung erforderlich sein.

F6 (C,S) SHOULD Kategorisierung der Informationen

Um dem Anwender zu ermöglichen nur die Informationen zu erhalten, die ihn interessieren sollte eine Kategorisierung des Inhaltsangebots des Servers möglich sein, anhand dessen der Anwender entscheiden kann ob Verbindung aufgebaut werden soll.

F7 (C) SHOULD Anpassungsfähigkeit hinsichtlich Systemeigenheiten

Da unterschiedliche Endgeräte unterschiedliche Funktionen zur Verfügung stellen sollte die Anwendung auf diese reagieren können.

F8 (C,S) SHOULD Synchrone und Asynchrone Kommunikation soll möglich sein

Direkte Interaktion erfordert eine Synchrone Verbindung. Das Anfrage-Antwort verfahren ist für den Austausch von Informationen zu bevorzugen, daher soll beides Möglich sein.

F9 (S) NICETOHAVE Globale Informationsdienste

Informations-Ticker, also Dienste die aktuelle Informationen an den Client weiterleiten können, z.B. Bundesligaticker, müssen darüber informiert werden, wie sie den Client erreichen

können. Die sich ständig verändernde Position des Clients muss diesen Tickerdiensten zugänglich gemacht werden.

4.2.3 Nichtfunktionale Anforderungen

NF1 (C) MUST Verbreitung

Um die Verbreitung und die Akzeptanz der Anwendung zu fördern muss sie auf einem Großteil von aktuellen Geräten einsatzfähig sein.

NF2 (C,S) MUST Mobilität

Die Anwendung muss mobil sein. D.h. z.B. kabellos kommunizieren können. Der Einsatz der Software darf den Anwender nur wenig bis gar nicht räumlich einschränken. Kommunikation muss gewährleistet sein, d.h. sowohl Server als auch Client müssen die Übertragungstechnologie einsetzen können.

NF3 (C,S) MUST Positionsinformationen

Ein ortsbasierter Dienst bietet Informationen für einen bestimmten Ortskontext. Es muss also bestimmbar sein, welcher Ortskontext aktuell ist.

NF4 (C,S) MUST Erweiterbarkeit

Clients und auch Server müssen auf Marktentwicklungen reagieren können. Neue Funktionalität sollte mit wenig Aufwand implementierbar sein.

NF5 (C,S) MUST Funktionalität in geschlossenen Räumen

Die Anwendung muss innerhalb von Gebäuden verwendbar sein.

NF6 (C,S) MUST Standardanforderungen an Software

Client und Server müssen den unter 1.2 genannten Standardanforderungen an qualitative Software entsprechen.

NF7 (C,S) SHOULD Betriebskosten

Anwendungen die im Betrieb Kosten verursachen zum Beispiel Internetgebühren sind unattraktiver für den Endverbraucher.

NF8 (C,S) SHOULD Initialkosten

Hohe Initialkosten, also Kosten die einmalig für den Betrieb aufgewendet werden müssen (Infrastruktur), verringern die Akzeptanz beim Benutzer.

NF9 (C) SHOULD Zusatzhardware

Zusatzhardware für den Grundbetrieb sollte nach Möglichkeit nicht benötigt werden.

NF10 (C,S) SHOULD Positionsgenauigkeit
Für den Einsatz dieser Art von ortsbasiertem Dienst ist eine gewisse Genauigkeit der Positionsdaten erforderlich. Daher darf die Abweichung von der tatsächlichen Position nicht mehr als zehn Meter betragen.

4.2.4 Einordnung

Die Liste beschreibt nur die Hauptanforderungen an die zu entwickelnde Software. Da die Konzeption von Software ein iterativer Prozess ist, stellt sie nur ein Zwischenergebnis dar. In diesem Fall bildet sie aufgrund des kleinen Zeitrahmens die Grundlage für das weitere Vorgehen.

5 Design

Im folgenden Abschnitt werden die grundlegenden Design Entscheidungen wie einzusetzende Technologien, Softwarearchitektur und Softwarebausteine sowie Probleme und mögliche Lösungen derer dargestellt und erörtert.

5.1 Entscheidungen

Die zur Erfüllung der Anforderungen getroffenen Design-Entscheidungen werden nun erläutert. Resultierend daraus ergeben sich Einschränkungen die in der Software-Architektur berücksichtigt werden müssen.

5.1.1 Positionierung

Essentiell wichtig für den Einsatz von ortsbasierten Diensten ist die Zusammenarbeit mit einem Ortungs-Dienst. Die Art des Ortungs-Diensts beeinflusst die Wahl der kompatiblen Endgeräte stark. In diesem Abschnitt wird erörtert, welche Anforderungen eine besonders wichtige Rolle spielen und wie diesen durch die richtige Auswahl der Mittel begegnet wird.

Anforderungen

Analysiert man die in Kapitel 3.1 genannten Positionsbestimmungstypen hinsichtlich ihrer Eignung zur Erfüllung der Anforderungen im Abschnitt 4.2 lassen sich schnell bestimmte Anforderungen ausmachen deren Erfüllung die Auswahl stark limitiert.

Gewährleistung der Funktionalität in geschlossenen Räumlichkeiten (NF5):

Um innerhalb von Gebäuden Positionsdaten ermitteln zu können, müssen diese mit einer entsprechenden Sensorik ausgestattet sein. Ein Einsatz von Satellitenpositionierung ist durch die Abschirmung des Signals durch die Wände in Gebäuden nur stark eingeschränkt möglich. Dementsprechend ist von einem Einsatz abzusehen.

Vermeidung von kostenintensiver Infrastruktur (NF8),

Simple Verbreitung der Anwendung (NF1),

Verzicht auf Zusatzhardware für Clients (NF9):

Die Erfüllung dieser drei Anforderungen macht den Einsatz von teurer, im Aufbau aufwändiger Positionierungshardware sowie den Einsatz von Zusatzmodulen für den Client unmöglich. Somit beschränkt sich die Auswahl der möglichen Systeme auf folgende Optionen:

- Positionierung anhand von GSM-Zellen.
- Visuelle Mustererkennung anhand von integrierter Kamera
- Nutzung von kostengünstiger Zusatzhardware zur Nutzung von Positionierungsdiensten
- Fingerprinting oder Annäherungsmessung durch WLAN-Module.
- Annäherungsmessung anhand des vielfach vorhandenen Bluetooth-Kurzstreckenfunk Moduls

Bereits ausgeschieden ist GPS-Navigation wegen Verstoßes gegen Anforderung NF5

Positionierung anhand von GSM-Zellen unterstützt nur eine Genauigkeit von maximal 40-50 Metern (Küpper, 2005). Anforderung NF10 ist somit nicht erfüllt.

Die Mustererkennung schränkt den Anwender in seiner Mobilität stark ein, da sie erfordert, dass ein direkter visueller Kontakt zwischen dem Endgerät und dem zu erkennenden Muster besteht. Problematisch ist weiterhin die hohe Leistungsanforderung die visuelle Mustererkennung stellt, Mobile Endgeräte können diese meist nur in sehr limitiert Maße erfüllen (siehe Abschnitt 3.3). Als Ausnahme sei der sogenannte KAYWA Reader (KAYWA, 2008) genannt, der über eine Art Barcodesystem Information visuell erfassen und verarbeiten kann.

Die Nutzung von WLAN-Modulen scheitert an der zurzeit geringen Verbreitung in mobilen Endgeräten.

Gegen die Nutzung von kostengünstiger Zusatzhardware spricht zur Zeit der Erstellung dieser Arbeit nur das nicht Vorhandensein dieser. Durch die erschwerte Erweiterbarkeit von mobilen Endgeräten ist Zusatzhardware schwierig zu integrieren.

Fazit

Positionsbestimmung durch Annäherungsmessung unter Zuhilfenahme der Bluetoothfunktechnik ist die Variante die den Anforderungen am Besten gerecht wird:

Kosten – Bluetooth-Hardware hat einen aktuellen Marktpreis von 5-10 € (Stand Sommer 2008). Es fallen keine Verbindungsgebühren an. Es muss allerdings beachtet werden, dass es sich hierbei um Erweiterungen für PCs handelt. Die Anschaffungskosten der PCs müssen ebenfalls berücksichtigt werden. Da in den vorgestellten Einsatzszenarien 4.1 meist eine entsprechende PC-Infrastruktur vorhanden ist, werden diese als vernachlässigbar angesehen.

Verbreitung und Zusatzhardware – Eine Majorität der aktuell auf dem Markt befindlichen Mobiltelefone unterstützen Bluetooth bereits nativ (Group, 2007).

Positionierungsgenauigkeit – Beschränkungen der Sendeleistung bzw. Nutzung von speziellen Antennen (Richtungsantennen, vgl. Abschnitt 5.3) ermöglichen Radien von zehn Metern und weniger.

5.1.2 Mobile Laufzeitumgebungen – Bewertung und Vergleich

Um der in Abschnitt 1.2 geforderten hohen Kompatibilität zu aktuellen Endgeräten gerecht zu werden und damit auch die Erfüllung der nichtfunktionalen Anforderung NF1 nach einer einfach zu verteilenden Anwendung positiv zu beeinflussen, wird die Software für eine Mobile Laufzeitumgebung implementiert werden (siehe Abschnitt 3.4). Im Vergleich zu Anwendungen die speziell für die proprietären Betriebssysteme der Mobilen Endgeräte (vgl. Abschnitt 3.3) entwickelt werden büsst man zwar Flexibilität hinsichtlich des Zugriffs auf Geräteeigenheiten ein, vervielfältigt aber die möglichen Einsatzgebiete enorm.

Der aktuelle Markt für Mobile Laufzeitumgebungen wird beherrscht von zwei Produkten, zum einen Sun Microsystems Java 2 Mobile Edition (J2ME, (Sun Microsystems, 2008e)) zum anderen Microsofts Dot Net Compact Framework (.NET CF, (Microsoft, 2008a)). Die jeweiligen Produktmerkmale werden in der folgenden Übersicht (Michael Juntao Yuan, 2003) dargestellt.

Leider ist es schwierig genaue aktuelle Zahlen für die Verbreitung der beiden Technologien zu finden. Anhand von Aussagen die in verschiedenen Web-Artikeln getroffen wurden lässt sich jedoch ein klarer Trend ermitteln. Die Anzahl von J2ME fähigen Geräten übersteigt die Anzahl der .NET CF Geräte bei weitem (vgl. (Tometa Software, 2006)).

Eigenschaft	.NET CF	J2ME
Geräteanforderungen	Leistungsstark, teuer	Leistungsschwach, billig
Kosten	Hoch	Mittel
Sprachunterstützung	C#, Visual Basic.NET	Java (Mobile Edition)
Plattform	Pocket PC, Windows CE	Alle mit Java 2 ME Virtueller Maschine
Zugriff auf Natives System	Per P/Invoke wo unterstützt	-
Spezifikations Prozess	Nur Microsoft	Gemeinschaftlich (JSR- Java Specification Request)
Aktuelle Verbreitung	Gering	Hoch

Tabelle 5.1: Vergleich mobile Laufzeitumgebungen

Vergleicht man weiterhin die Leistungsmerkmale der beiden Kontrahenten lässt sich feststellen, dass abgesehen von dem fehlenden nativen Zugriff, also z.B. direktem Hardware-Zugriff, keine ausschlaggebenden Unterschiede festgestellt werden können.

Direkte Gegenüberstellung mit Bewertung von Einzelmerkmalen

Eine kurze Zusammenfassung der entscheidenden Merkmale, abgeleitet aus den nichtfunktionalen und funktionalen Anforderungen, gewichtet analog zu der jeweiligen Anforderung.

Eigenschaft	.NET CF	J2ME	Gewichtung/Anforderung
Verbreitung / Kompatibilität	- -	+ +	Must / 4.2.3
Zugriff auf drahtlos Kommunikation	+	+	Must / 4.2.3
Kosten	O	+	Should / 4.2.3
Erweiterbarkeit	O	+	Should / 4.2.3
Darstellung von Inhalt	+	O	Must / 4.2.2

Tabelle 5.2: Vergleich von mobilen Laufzeitumgebungen

Verbreitung / Kompatibilität

Viele Geräte auf denen die .NET CF Laufzeitumgebung läuft, unterstützen auch die Virtuelle Maschine von J2ME. Da diese eine asymmetrische Beziehung ist, gilt das Gegenteil leider nicht. Weiterhin einschränkend wirkt die Hardware-Anforderung die .NET CF an das

Mobile Gerät stellt, J2ME ist hier, durch seine Konfigurierbarkeit klar im Vorteil (vgl. (Tometa Software, 2006).

Zugriff auf Drahtloskommunikation

Beide Umgebungen ermöglichen den Zugriff auf die jeweilige Hardware über spezielle Schnittstellen-APIs. Daher ergibt sich kein Unterschied in der Bewertung.

Kosten

Da die Laufzeitumgebungen selbst kein Kostenfaktor darstellen, bezieht sich diese Bewertung allein auf die Hardwarekosten wodurch das .NET CF durch seinen Leistungshunger negativer bewertet werden muss. Zusätzlich zu den Hardwarekosten könnten Kosten gewertet werden die durch die Inanspruchnahme einer Certificate Authority zur Signierung der Software entstehen. Diese entstünden allerdings in beiden Fällen und sind somit zu vernachlässigen.

Erweiterbarkeit

Hinsichtlich Erweiterbarkeit der Softwarebasis unterliegt J2ME dem sogenannten Java Community Process (JCP). Ein von mehreren führenden Industrievertretern geleiteter Prozess der Vorschläge sogenannte Java Specification Requests (JSR) macht und im öffentlichen Dialog über deren Umsetzung entscheidet (Sun Microsystems, 2008a).

Beim .NET CF ist dieser Prozess alleinig Microsoft vorbehalten, nur mit indirekter Beteiligung von weiteren Unternehmen (Michael Juntao Yuan, 2003). Auch wenn dieses „Monopol“ Vorteile bringt bei der Geschwindigkeit mit der eine Änderung propagiert werden kann, muss die fehlende Einwirkungsmöglichkeit von Außen negativ bewertet werden.

Darstellung von Inhalt

Das .NET Compact Framework ist von Haus aus auf eine leistungsfähigere Hardware angewiesen, diese ist meist auch mit besseren Ausgabemöglichkeiten ausgestattet, daher die positive Bewertung.

Fazit

Der Anteil und die Qualität der Erfüllung der unter Abschnitt 4.2 aufgestellten Anforderungen spricht, wie die vorhergegangene Analyse zeigt, eindeutig für die Nutzung von J2ME als Implementierungsplattform. Somit wird J2ME für die Umsetzung der Client-Komponente verwendet werden.

5.1.3 Mobile Laufzeitumgebung J2ME

J2ME auch bekannt als JME (Java Micro Edition) steht für Java 2 Platform Micro Edition, es ist ein mobiles Pendant zur Desktop Version J2SE – Java 2 Platform Standard Edition. Dieses Kapitel gibt kurz einen Überblick über den Aufbau, die Funktionsweise und Entwicklung von J2ME. Zudem wird die Eignung für dieses Projekt analysiert und belegt.

Ursprung

Laut (Sun Microsystems, 2008d) war die Hauptintention für die Entwicklung von J2ME, eine Umgebung zu schaffen, die den Anforderungen „kleiner“ Geräte gerecht wird:

- wenig virtueller Speicher
- geringe Prozessorleistung
- eingeschränkte Darstellung durch kleine Displays.

J2ME Bestandteile

Um die Vielzahl verschiedener Geräte möglichst optimal unterstützen zu können, wurde J2ME in verschiedene Komponenten eingeteilt die jeweils unterschiedliche Anforderungen stellen. Einen Überblick über unterstützte Geräte erhält man unter „The Java Me Device Table“ (Sun Microsystems, 2008c).

Konfigurationen

Die Grundlage bilden sogenannte Konfigurationen. Eine Konfiguration schreibt vor, welche Eigenheiten der Virtuellen Maschine vorhanden sein müssen und bestimmt eine Basis an notwendigen Bibliotheken. Bis heute haben sich zwei Konfigurationen durchgesetzt. Connected Device Configuration (CDC) und Connected Limited Device Configuration (CLDC). Das Hauptunterscheidungsmerkmal dieser beiden Konfigurationen ist, abgesehen von ihrer Implementierung, das jeweilige Einsatzgebiet.

CDC ist kompatibel mit der J2SE Umgebung und somit nur bedingt optimiert für Geräte mit den oben genannten Einschränkungen, mögliche Einsatzgebiete sind zum Beispiel leistungsfähige SmartPhones oder Settop-Boxen. Sun selbst schreibt der Hauptunterschied liege in der angestrebten Speicherlast, CDC sei entwickelt für Systeme mit ungefähr zwei Megabyte and RAM und zweieinhalb Megabyte ROM Speicher. CLDC hingegen ziele auf

Geräte ab mit einer RAM Kapazität von 128 KB bis 512 KB. Daher sei eine Kompatibilität zur Desktop Version (J2SE) nicht zu erreichen (Sun Microsystems, 2008b).

Da Smart-Phones und Handhelds aktuell nur vergleichsweise gering verbreitet sind (Statistisches Bundesamt, 2007a) und somit einer Hauptanforderung widersprechen, wird auf eine Erläuterung von CDC verzichtet.

CLDC 1.0a

Die auf eingeschränkte Geräte abgezielte Connected Limited Device Configuration (CLDC, JSR-30 (Sun Microsystems, 2000)) wurde im Jahre 2000 in der Version 1.0a veröffentlicht

Als Ziele für die CLDC wurden folgende Charakteristiken spezifiziert:

- **160kB bis 512kB für Java nutzbarer Speicher**
  - 128kB nicht flüchtiger Speicher für die Java und CLDC Bibliotheken
  - Mindestens 32kB flüchtiger Speicher für den Betrieb
- Geringe Leistungsaufnahme, häufiger Batteriebetrieb.
- Ein 16 oder 32 Bit Prozessor
- **Konnektivität zu einem Netzwerk**

Um das vorgegebene Speicherlimit zu erreichen wurden verschiedene Klassen verkleinert oder aber komplett weggelassen. Eine Liste der verfügbaren Klassen sowie der vorgenommenen Änderungen kann man in der CLDC-Spezifikation (Sun Microsystems, 2007) einsehen.

Ein besonderes Augenmerk bei der Spezifikation der CLDC wurde auf die Konnektivität gelegt. Es wurde ein auf geringen Ressourcenbedarf abgestimmte Klassensystem entwickelt, das sogenannte Generic Connection Framework (GCF). Dies bietet Entwicklern Basisklassen zur Implementierung von verschiedenartigen Kommunikationsmethoden. Unterschieden werden diese durch einen eindeutigen Prefix im Verbindungsstring, z.B. File:// für einen Zugriff auf das Dateisystem über das JSR-75-Profil (Sun Microsystems, 2004).

Hierbei zu beachten ist, dass alleinig die Grundstruktur bereit gestellt wurde. Spezielle Implementierungen wie HTTP oder Sockets sind wegen der Konfigurationsdefinition (JCP, 2003) nicht Bestandteil des CLDC sondern ausgelagert in weitere J2ME-Profile.

CLDC 1.1

Die Version ist abwärtskompatibel zur Version 1.0 und bietet nur geringfügige Änderungen. Für Entwickler von besonderem Interesse ist die hinzugekommene Unterstützung von Fließkomma Operationen und Datentypen. Diese wurden von Version 1.0a nicht unterstützt.

Damit verbunden wurde die Speicherzielsetzung von mindestens 160kB auf 192kB angehoben. Weitere Änderungen können der CLDC 1.1 Spezifikation (JCP, 2003) entnommen werden.

Profile

Ein J2ME Profil beschreibt eine Anzahl APIs die die von der Basiskonfiguration bereitgestellte Funktionalität erweitern. Dies geschieht vorrangig zur Spezialisierung auf eine bestimmte Zielplattform. Das im Zusammenhang mit CLDC meist genutzte Profil ist das Mobile Information Device Profile (MIDP, (JCP, 2000)).

MIDP (Mobile Information Device Profile)

In der Spezifikation von MIDP wird das mit dem Profil verfolgte Ziel wie folgt beschrieben:

“The goal of this specification is to define the architecture and the associated APIs required to enable an open, third-party, application development environment for mobile information

devices, or MIDs.“ (JCP, 2000)

Die Zielplattformen dieses Profils sind somit sogenannte Mobile Information Devices (MID). Ein MID wird beschrieben als Gerät mit mindestens folgenden Hardware-Eigenschaften (JCP, 2000):

- Anzeige
  - Auflösung 96x54
  - Farbtiefe 1-bit (Schwarz/Weiß)
  - Anzeigeverhältnis ungefähr 1:1
- Eingabemöglichkeit
  - Mindestens eines der Folgenden: „Ein-Hand Keyboard“, „Zwei-Hand Keyboard“, Touchscreen.
- Speicher
  - 128 kB nicht flüchtiger Speicher für die MIDP Bestandteile
  - 8 kB nicht flüchtiger Speicher für persistente Anwendungsdaten wie z.B. Einstellungen
  - 32 kB flüchtiger Speicher für die J2ME Laufzeitumgebung
- Netzwerk

  - Zweiwege drahtlos Kommunikation mit limitierter Bandbreite

Zusätzlich zu den Hardwareanforderungen muss ein MID auch weitere Einschränkungen hinsichtlich seiner Software erfüllen um dem MIDP zu entsprechen. Diese Anforderungen dienen der spezifizierten Grundfunktionalität, also Beispielsweise die Ansteuerung eines Bitmap-Displays oder auch die Fähigkeit die oben genannten Eingabemethoden zu verwenden. Die vollständige Liste kann der (JCP, 2000) entnommen werden. Zusammengefasst lässt sich feststellen, das MIDP bietet APIs für mindestens folgende Bereiche (JCP, 2000):

- Anwendung
  - Wie wird eine MIDP-Anwendung (MIDlet) erstellt/gestartet/beendet. Wie verhält sie sich.
- Benutzeroberfläche
  - Darstellung von Informationen inklusive Verarbeitung von Benutzereingabe.
- Persistente Speicherung von Daten
  - Speichert Daten über die Laufzeit vom MIDlet hinaus, Record Management System (RMS) (JCP, 2000)
- Netzwerkverwaltung
- Timer
  - Funktionalität zur intervallgesteuerten oder geplanten Ausführung von Code.

Das Mobile Information Device Profile ist das aktuell meist genutzte Profil für Anwendungen auf J2ME-Geräten (meist Mobiltelefone). Es bietet die Funktionalität die für Informationsdarstellende und –verarbeitende Anwendungen essentiell sind.

Optionale Pakete

Der dritte und letzte Bestandteil von J2ME sind die sogenannten optionalen Pakete. Sie definieren APIs für die Nutzung einer oder mehrere Technologien. Da sie optional sind bedeutet die Nutzung eine Beschränkung der Portabilität der Anwendung. Daher sollten optionale Pakete nur eingesetzt werden, wenn ihre Funktionalität definitiv benötigt wird. Im Folgenden werden nun die Pakete vorgestellt, die für den Betrieb der zu entwickelnden Anwendung unmittelbar erforderlich sind. Dabei ist zu beachten, dass der Einsatz eines Paketes direkt auch ein Vorhandensein der jeweiligen Technologie erfordert und somit in direktem Zusammenhang mit den Anforderungen steht.

Kommunikation und Positionsbestimmung

JSR 82: JavaTM APIs for Bluetooth (Sun Microsystems, 2006)

Die Java APIs for Bluetooth wireless technology (JABWT) bieten einen „high level“-Zugriff auf einen Großteil der Funktionalität des Bluetooth Kurzstreckenfunk Transceivers. Sie sind eine Implementierung des von der CLDC spezifizierten Generic Connection Frameworks (GCF), dies macht sie einsetzbar in allen Anwendungen, die das CLDC oder das GCF als Grundvoraussetzung haben. In Desktop-Umgebungen, also J2SE oder J2EE Anwendungen, kann JABWT durch einbinden der durch JSR 197 (Generic Connection Framework Optional Package for the J2SE^{TM} Platform (Sun Microsystems, 2003)) spezifizierten Funktionalität eingesetzt werden.

Ursprünglich wurde JSR 82 für J2ME Anwendungen konzipiert, da es die JCP-Experten als wahrscheinlich angesehen haben, dass die ersten Anwendungen mit Bedarf für Bluetooth Konnektivität aus dem Bereich der Mobilen Endgeräte stammen würden.

Voraussetzungen

Ähnlich der bereits erwähnten Konfigurationen hat auch JABWT spezifische Anforderungen an die darunterliegende Hard- und Software.

Hinsichtlich J2ME äußert sich dies beispielsweise durch einen erhöhter Speicherbedarf von 512 kB für die JABWT Klassen. Weiterhin ist natürlich ein Bluetooth Transceiver mit zugehörigem Bluetooth Stack erforderlich. Dieser muss mindestens, die in der BT-Spezifikation 1.1 definierten Protokolle SDP, RFCOMM und L2CAP unterstützen und mindestens den folgenden BT-Profilen entsprechen, GAP, SDAP, SPP.

Weiterhin wird ein sogenanntes Bluetooth Control Center (BCC) benötigt. Das BCC bietet Zugriff auf im Gerätekontext getroffene Einstellungen für den BT-Transceiver, wie den „Benutzerfreundlichen Gerätenamen“ oder die Sichtbarkeit. Zudem synchronisiert das BCC die Änderung dieser Einstellungen und ist für das Auflösen von Konflikten bei parallelem widersprüchlichem Zugriff. Eine Anwendung möchte, dass der Transceiver nicht sichtbar ist, die andere erfordert volle Sichtbarkeit) zuständig. Das BCC ist meist im Betriebssystem des MEs implementiert.

Funktionalität

Aus den Vorraussetzungen lassen sich folgende Funktionalitäten ableiten:

SDP:

- Dienste registrieren
- Finden von Geräten und Diensten

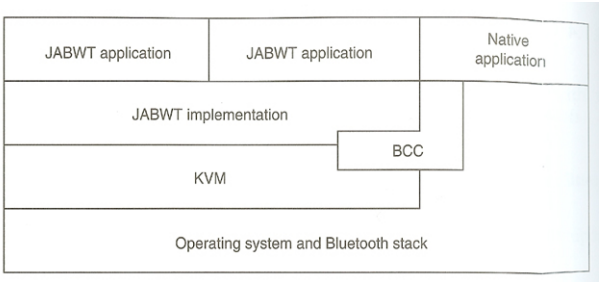


Abbildung 5.1: JABWT Schichtung

RFCOMM, L2CAP, SPP:

- Direkte Protokollgesteuerte Verbindungen aufbauen/verwenden

BCC

- Sicherung der Verbindungen
- Steuerung des gemeinsamen Zugriffs auf den Transceiver

Eignung

JABWT bietet direkten Zugriff auf das Bluetooth-RFCOMM-Protokoll und gewährleistet damit gekapselt durch das GCF eine Zweiwege Kommunikation per Stream. Damit wird ein drahtloser Austausch von Informationen ermöglicht. Dies erfüllt die nichtfunktionale Anforderung NF2 ohne Einschränkungen.

Der von JABWT bereitgestellte Zugriff auf das Service Discovery Protocol gibt der Clientanwendung die Möglichkeit, einen Dienst zu registrieren der prüfende Servern über das Vorhandensein der Software informiert. Somit lassen sich alle benötigten Orts-Informationen erhalten und verarbeiten.

Persistentes Dateisystem

JSR 75: PDA Optional Packages for the J2ME Platform (Sun Microsystems, 2004)

Das unter dem Java Specification Request 75 entwickelte Profil für CLDC und CDC Geräte enthält zwei Hauptbestandteile. Zum einen die „FileConnection-API“ und zum anderen die sogenannte „PIM-API“. Die Abkürzung PIM steht hierbei für *Personal Information Management.* Die Spezifikation (JCP, 2001) beschreibt PI-Daten als Informationen die in Adressbüchern, Kalendern oder TODO-Listen vergefunden werden. Die Organizer-Funktionalität die mit dieser Art Daten einhergeht wird von vielen Mobiltelefonen unterstützt.

Die FileConnectionAPI stellt eine generische Schnittstelle für das auf dem Mobiltelefon vorhandene Dateisystem dar. Sie bietet die Möglichkeit sowohl schreibend als auch lesend, zum

Beispiel auf etwaig vorhandene Speicherkarten, zuzugreifen. Der Zugriff wird, sofern von der Implementierung vorgesehen, über die Nutzung des GCF abgewickelt. Der Zugriff auf das Dateisystem ermöglicht die Implementierung einer Caching-Infrastruktur zur Schonung des flüchtigen Speichers.

Fazit

Zusammenfassend kann festgestellt werden, J2ME eignet sich nahezu ohne Einschränkungen für die Umsetzung der angestrebten Software. Somit erweitern sich die Implementierungsbeschränkungen um folgende Punkte:

- Implementierungssprache ist JAVA
- Der Client soll als MIDlet entwickelt werden und muss die CLDC unterstützen
- Die Kommunikation von Client und Server soll über Bluetooth unter Zuhilfenahme von JABWT erfolgen.

5.1.4 Informationsformat

Das unter Abschnitt 3.7.1 vorgestellte XHTML Mobile Profile bietet zusammen mit der WML2 Spezifikation das erfolgversprechende Format für die Darstellung von Informationen. Da die Implementierung eines der Spezifikation entsprechenden Webbrowsers den Rahmen dieser Arbeit bei weitem übersteigt, wird vorerst auf den Einsatz verzichtet. Im Gegenzug wird auf ein ebenfalls XML basiertes Informationsformat gesetzt, welches die für die beispielhafte Darrstellung der Funktionalität der Anwendung nötigen Elemente unterstützt. Denkbar wäre eine Konversion von XHTML Mobile in das proklamierte Format.

5.1.5 Server

Da auf den Mobilgeräten eine Java Anwendung zum Einsatz kommen wird, bietet es sich an, auch auf dem Stationärteil auf Java zu setzen. Bereits entwickelte Kommunikationsmodule ließen sich so sowohl auf Client als auch auf Server einsetzen. Dies ist natürlich nur möglich, wenn Server- und Client-Javaumgebung die vom Kommunikationsmodul verwendeten Komponenten unterstützen. Das Kommunikationsmodul wird daher nur auf Basiskomponenten wie java.io.input- und java.io.outputstreams setzen und auch sonst nur Bestandteile der Schnittmenge von J2ME und J2SE einsetzen.

Kommunikation mit dem Client

Da Bluetooth als Kommunikationsmedium festgelegt wurde, ergibt sich eine weitere Voraussetzung für den Einsatz von J2SE als Serverlaufzeitumgebung. Nämlich die Möglichkeit der Nutzung dieser Technologie. Wie bereits im Abschnitt 5.1.3 beschrieben bietet der JSR-197 die Möglichkeit des Einsatzes vom Generic Connection Framework in J2SE-Umgebung. Das Generic Connection Framework ist wie der Name schon sagt generisch, es fehlt also noch eine Implementierung des JSR-82 um Bluetooth einsetzen zu können. Diese Funktionalität bietet BlueCove (BlueCove Team, 2008), eine Open-Source-Java-Bibliothek zur Nutzung von verschiedenen Bluetooth Stacks (WIDCOMM, WinSock, BlueSoleil) auf unterschiedlichen Betriebssystemen (Windows, Mac OS, Linux)

Kommunikation mit anderen Servern

Der in der Anforderungsanalyse unter Kapitel 4.2.2 beschriebenen Anforderung nach einem globalen Verarbeiter von Clientanfragen lässt sich am besten mit einer weiteren Implementierung eines verteilten Systems begegnen. Um gewährleisten zu können, dass die Server untereinander bekannt sind und kommunizieren können könnte auf eine Middleware gesetzt werden. Die Einordnung der Anforderung als Nice-to-have legt allerdings nahe, zum Zwecke der Zeitersparnis vorerst auf eine Implementierung zu verzichten.

Fazit

Festzuhalten ist also, für die Implementierung des Servers wird Java verwendet. Das ermöglicht die Wiederverwendung des Kommunikationsmoduls. Auf physikalischer Ebene wird Bluetooth verwendet, dass über die BlueCove Bibliothek verfügbar gemacht wird. Auf die Nutzung einer Middleware für die Server-Server-Kommunikation wird aufgrund Zeitmangels vorerst verzichtet.

5.1.6 Kommunikation

Ein Hauptbestandteil der zu entwickelnden Anwendung ist der Informationsaustausch zwischen Client und Server. Um Informationen übermitteln zu können muss zwischen den beiden Komponenten eine Kommunikationstruktur etabliert werden. Bei der Wahl der geeigneten Methode beziehungsweise Technologie müssen die zuvor bereits getroffenen Entscheidungen beachtet werden:

- Als Plattform für den Client kommen Mobiltelefone zum Einsatz.

- Bluetooth wird für die Positionsbestimmung eingesetzt.

Auch wenn Positionierung und Kommunikation auf den ersten Blick nicht miteinander korrelieren, ist die Wahl von Bluetooth doch von erheblicher Bedeutung. Aus der Konjunktion der beiden Aussagen lässt sich schließen, dass die Mobiltelefone Bluetooth unterstützen müssen. Somit kann es ebenfalls als Methode zur Datenübertragung eingesetzt werden.

Vergleich

Die Palette der möglichen Übertragungstechnologien, die bereits im Grundlagen-Kapitel 3.3 benannt worden ist, soll auf Eignung für das zu entwickelnde Projekt geprüft, dabei werden die jeweiligen Kategorien entweder mit einem „+“ für gut geeignet, „o“ für Wertneutral oder einem „-“ für weniger gut geeignet bewertet. Die zu bewertenden Kategorien und der Gewichtung leiten sich aus den Anforderungen ab:

Reichweite:

Die Reichweite hat direkten Einfluss auf die Nutzbarkeit der Anwendung. Je Höher sie ist, desto besser. Für eine Neutrale Wertung müssen mindestens 5 Meter erreicht werden, für eine Positive Wertung mehr als 20 Meter.

Kosten:

Kosten die bei der Kommunikation anfallen haben einen negativen Einfluss auf die Anwendermeinung. Das bedeutet je geringer die Kosten desto besser. Somit ist eine binäre Wertung zu verwenden, „-„ für anfallende Kosten und „+“ für kostenfreie Übertragung.

Verbreitung:

Die Verbreitung der Technologie bei aktuellen Mobiltelefonen. Der Verbreitungsgrad ist proportional zu der Anzahl potentieller Anwender. Da eine Hauptanforderung die Maximierung der möglichen Nutzer ist (NF1), ist diese Kategorie Wertidentisch mit der Summe der beiden anderen.

Technologie/ Kategorien	Reichweite (20%)	Kosten (30%)	Verbreitung (50%)
GSM	+	-	+
WLAN	+	+	-
Bluetooth	o	+	+
IRDA	-	+	+

Tabelle 5.3: Bewertung Technologien

Fazit

Aus der Tabelle lässt sich ablesen, dass sich Bluetooth als am Besten geeignete Technologie darstellt. Einschränkend zu bemerken ist die im Vergleich geringere Übertragungsgeschwindigkeit, da aber Mobile Endgeräte nur geringe Speicherressourcen haben ist davon auszugehen, das nur geringe Datenmengen zu übertragen sind und dieser Faktor nur wenig ins Gewicht fällt. Für die Kommunikation von Client und Server in der Beispielanwendung wird also Bluetooth eingesetzt.

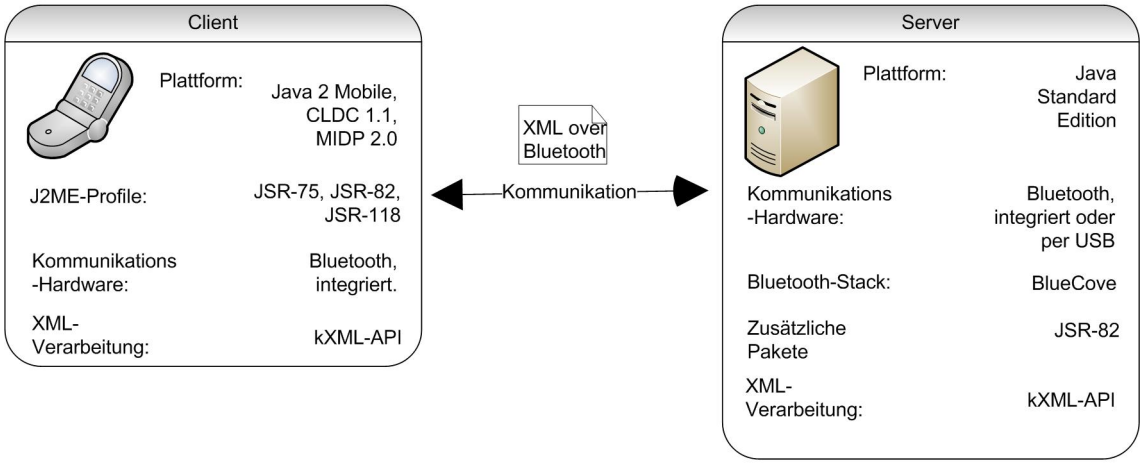


Abbildung 5.2: Designentscheidungen im Überblick

5.2 Vision

Nach den im vorherigen Kapitel getroffenen Designentscheidungen wird in diesem Abschnitt eine sich daraus ergebende Anwendungsbeschreibung vorgestellt. Diese beschreibt auf hoher Abstraktionsebene grob die Bestandteile und deren Zusammenwirken.

5.2.1 LBS Supply Chain

Aus der im Kapitel Grundlagen LBS beschriebenen LBS-Supply-Chain-Sicht (3.5.4) ergibt sich das Folgende Bild: Die Clientanwendung vereint Target und im übertragenen Sinne

auch LBS-User in sich. Die weiteren Rollen, Position-Originator, Location-Provider, Content-Provider und LBS-Provider werden allesamt vom Serverteil übernommen.

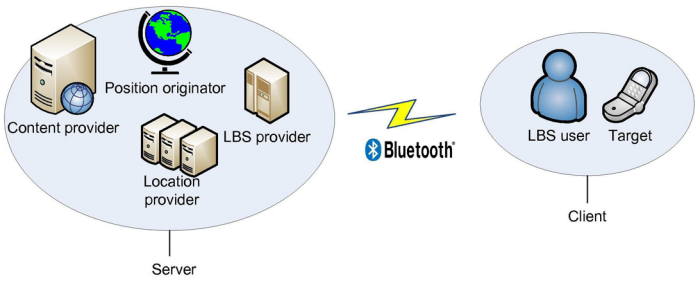


Abbildung 5.3: LBS Supply Chain

5.2.2 Anwendungsbeschreibung

Hinsichtlich der Funktion der Bestandteile ist somit folgendes Festzustellen. Der Client, ein CLDC-fähiges Mobiltelefon mit Bluetoothadapter, wird sobald er sich in Reichweite befindet, von einem Server aufgefasst und per Bluetooth über dessen Anwesenheit unterrichtet. In der Client-Anwendung erscheint der Server als Eintrag in einer Liste. Eine MIDP-Anwendung bildet die Basis für die Informationsdarstellung und die weitere Funktionen. Nach Aktivierung des Servereintrags baut die Clientanwendung eine serielle Bidirektionale Bluetoothverbindung mit dem Server auf und fordert eine Standard Datei an. Der Server reagiert und stellt die gewünschte Information über die bestehende Verbindung bereit. Die Client-Anwendung interpretiert die erhaltene Information und stellt diese dar, ermöglicht damit dem Anwender weitere Funktionalität des Servers in Anspruch zu nehmen.

5.3 Problemanalyse

Im Folgenden werden Probleme und Risiken erörtert die sich durch die Nutzung der gewählten Komponenten ergeben bzw. ergeben könnten.

5.3.1 Bluetooth

Die Rolle die Bluetooth in der zu entwickelnden Anwendung spielt ist eine sehr entscheidende. Sowohl die Positionierung als auch die Kommunikation werden darüber abgewickelt. Daher ist eine Analyse hinsichtlich möglicher Probleme sinnvoll.

Positionierung

Bluetooth als Positionierungsdienst einzusetzen birgt einige Nachteile. In Grafik 5.4 ist ein typisches Anwendungsszenario abgebildet. Der Einzugsbereich, also der Bereich in dem ein Client vom Server erkannt wird ist als Ellipse dargestellt. Wie sich nun unschwer erkennen lässt treten eine Vielzahl Schnittmengen auf, in denen ein Client von mehreren Servern erkannt werden würde. Das mag in geringem Ausmaße noch akzeptabel sein, im Extremfall führt dieses Verhalten aber dazu, dass dem Client an jeder Position alle Dienste zur Verfügung stehen. Für den Anwender ergibt sich somit ein sehr unübersichtliches Bild. Der Vorteil der Kontextbezogenheit von ortsbasierten Diensten geht verloren wenn man einen Dienst nutzt die Auswirkungen aber aufgrund der Entfernung nicht mehr wahrnehmbar sind. Dies betrifft vor allem Dienste, die die Clientanwendung zur Steuerung von weiteren Anzeigegeräten nutzen wollen, beispielsweise Anzeigetafeln auf Flughäfen oder Bahnhöfen.

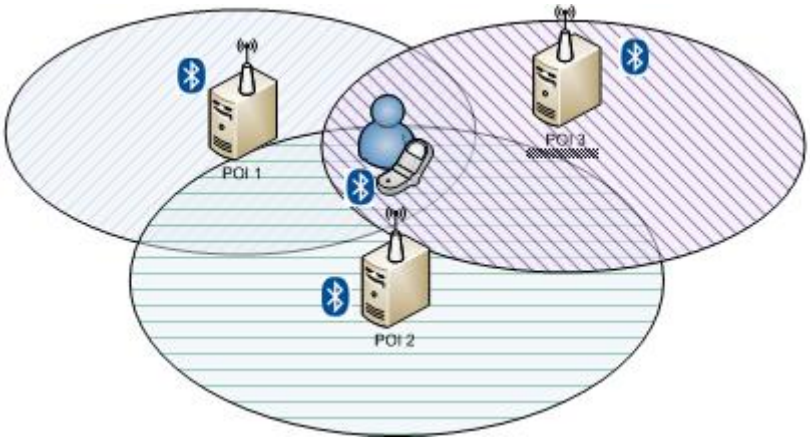


Abbildung 5.4: Einzugsbereich-Schnittmengen

Problematik

Die Hauptursache für die oben genannten Schwierigkeiten ist die Überlappung der Einzugsbereiche. Die dafür entscheidenden Kriterien sind die Sendeleistung und die Form der genutzten Antenne. Klasse Eins Bluetoothgeräte senden mit einer Maximalen Leistung von 100mW. Bei optimalen Bedingungen bedeutet das eine Reichweite von circa 100 Metern.

Die Standardisiert verbaute Isotropische Antenne strahlt in alle Richtungen gleichstark ab. Daraus ergibt sich ein Kugelförmiger Empfangsbereich, bei dem die Antenne im Ursprung steht.

Lösungsansatz

Um ein möglichst heterogenes Verteilungsmuster der Einzugsbereiche zu erreichen, das bedeutet das Vermeiden von Schnittmengen, kann entweder die Sendeleistung verringert oder

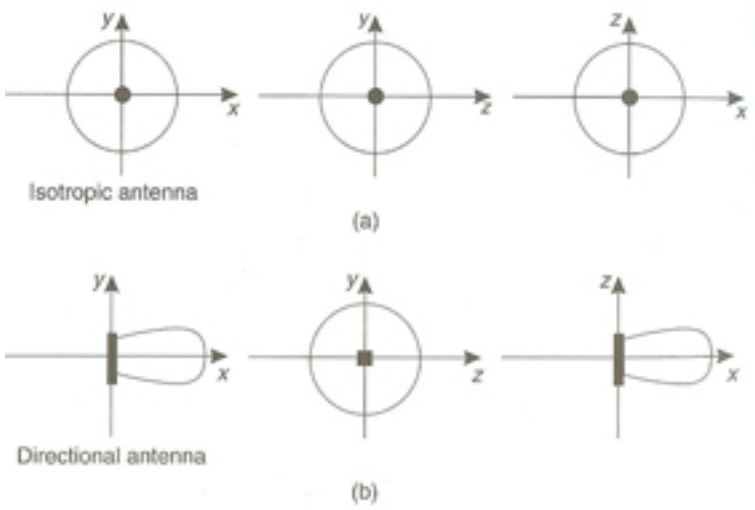


Abbildung 5.5: Antennenstrahlungsbereiche

aber die Form der Antenne verändert werden. Eine Verringerung der Sendeleistung geht meist mit einer Verschlechterung der Übertragungsrate einher, da sich der Signal-Rausch Abstand verringert und Nutzdaten häufig mehrfach übertragen werden müssen. Konzentriert man allerdings den Abstrahlungsradius mit einer direktionalen Antenne so entfällt der Zwang die Leistung zu verringern. Die Strahlungscharakteristik einer direktionalen Antenne ist vergleichbar mit der eines Scheinwerfers, ein leicht konisch zur Quelle zulaufender zylindrischer Strahlungsbereich. Um möglichst getrennte Einzugsbereiche zu erhalten, ließen sich also direktionale Antennen an der Decke montieren um das zu erfassende Areal zu „bestrahlen“ und zu erfassen.

Kommunikation

Neben der, bei geringen Datenmengen zu vernachlässigenden, geringen Übertragungsgeschwindigkeit können die Restriktionen der Bluetoothardware weiter Probleme aufwerfen. Bluetooth wurde ursprünglich entwickelt, um Peripheriegeräte kabellos an Mobiltelefone anzuschließen. Die Anzahl von gleichzeitig angeschlossenen Zusatzgeräten steigt nur in den seltensten Fällen über drei oder vier, somit wurde die Grenze maximal anschließbarer Geräte auf sieben festgelegt. Dies spiegelt sich auch in der Bluetootpaketstruktur wieder: Die Größe der sogenannten „Active Member Address“ im Headerteil beträgt drei Bit, also Acht mögliche Wertebelegungen inklusiv einer Broadcast-Adresse. Sie bestimmt den Empfänger des Pakets in einem Piconetz (Muller, 2001). Ein Piconetz kann also aus maximal Acht Teilnehmern bestehen, einem Bluetooth-Master und sieben Bluetooth-Clients.

Problematik

Bei der zu erwartenden hohen Nutzungsfrequenz ist das Verbindungslimit schnell erreicht.

Lösungsansatz

Um zu vermeiden, dass es zu Blockaden kommt, können verschiedene Ansätze verfolgt werden. Es ist möglich beim initiieren der Verbindung festzulegen, welcher Teilnehmer der Master der Bluetoothverbindung ist. Es bietet sich also an die Clientanwendung mit der Aufgabe der Verbindungssteuerung zu betrauen. Obwohl ein Mastergerät nur maximal sieben Clientgeräte bedienen kann, kann es währenddessen in mehreren anderen Piconetzen als Client partizipieren. Das bedeutet auch wenn die Serveranwendung bereits mit sieben Clientanwendungen kommuniziert, könnte sie weiterhin Verbindungen annehmen.

Eine zweite Möglichkeit ist die Ausstattung des Servers mit mehreren Bluetoothgeräten. Diese müssten dann bei drohender Überfüllung durch Bekanntgabe während der Bekanntmachung mit den Clientanwendungen genutzt werden. Zudem ließe sich durch die Art der Verbindung, das Anforderung-Antwort-Verfahren, die Intervalle der Clientverbindung auf ein Minimum zu reduzieren.

Problematik

Während ein Bluetoothgerät nach weiteren Geräten sucht, ist es für andere Geräte weder auffindbar noch kontaktierbar. Da ein Suchvorgang bei hoher Anzahl Bluetoothgeräte viel Zeit in Anspruch nimmt, ist der Server währenddessen blockiert.

Lösungsansatz

Für dieses Problem gibt es verschiedene Lösungsansätze:

- Die Server könnten mit zwei Bluetoothgeräten ausgestattet werden. Eines nur für die ständige Suche nach kompatiblen Clients, das Andere für den Transfer von Informationen und die Kontaktaufnahme.
- Die Clients könnten die Suchfunktion übernehmen. Das heißt, dass Sie in bestimmten Zeitabständen nach Servern mit dem jeweiligen Dienst suchen.
- Um Zeit zu sparen gibt es verschiedene proklamierte Lösungsansätze für effektivere Suchprozesse (Handurukande u. a., 2005). Diese erfordern allerdings jedoch meist einen Low-Level Zugriff auf die Stack-Implementation. Dies ist über Java aktuell nicht machbar.

Sicherheit

Daten die drahtlos übertragen werden sind naturgemäß einem hohen Risiko ausgesetzt, abgefangen oder verändert zu werden. Einmal entschlüsselt bieten sie zudem die Möglichkeit ein Bewegungsprofil des Anwenders zu erstellen. Bluetooth bietet von Haus aus eine Verschlüsselungsmethode an, diese erfordert es für jede Verbindung mit einem neuen Bluetoothgerät eine Schlüsselsequenz sowohl auf Server wie auch auf Client einzugeben beziehungsweise zu bestätigen. Hinsichtlich der zu entwickelnden Anwendung bedeutete dies für jede Verbindung zu einem neuen Server müsste ein manueller Eingriff erfolgen. Dies beeinflusst die Useability sowohl für Anwender als auch Serverbetreiber stark negativ.

Lösungsansätze

Alternativ zur Verschlüsselung auf der Bluetooth-Ebene kann eine Verschlüsselung der Nutzungsdaten gewählt werden. Dies ermöglicht es, durch Nutzung von Asynchronen Verschlüsselungsverfahren, öffentliche Schlüssel zur Laufzeit oder bereits bei der Kompilierung in die Anwendung zu integrieren und so dem Missbrauch vorzubeugen. Eine Erstellung von Bewegungsprofilen wird dadurch aber nicht verhindert. Weitere Informationen zum Thema Sicherheit von Bluetooth-Geräten finden sich in (Bundesamt für Sicherheit in der Informationstechnik, 2003).

6 Entwurf

6.1 Architektur

Die Architektur einer Anwendung ist der entscheidende Faktor hinsichtlich der Erfüllung von Qualitätsanforderungen. Sie bildet die Basis für den gesamten Entwicklungsprozess. In diesem Kapitel werden die Grundzüge der Architektur sowie deren Komponenten und deren Zusammenwirken beschrieben.

6.1.1 Einteilung

Auf Grund der beschränkten Leistungsfähigkeit von mobilen Endgeräten ist es nötig, diese nach Möglichkeit nur wenig zu belasten. Die einfachste Möglichkeit dies zu erreichen, ist eine Aufteilung der Verantwortlichkeiten, das bedeutet, zur Entlastung wird das ME nach Möglichkeit nur zur Anzeige von Inhalt verwendet.

Die Client Anwendung auf dem ME wird im Folgenden als **Client**, das Gegenstück als **Server** bezeichnet. Zusätzlich zu den konzeptionell getrennten Bestandteilen wird, da es sich um eine verteilte Architektur handelt, ein gemeinsames **Kommunikationsprotokoll** benötigt.

6.1.2 Client-Server-Verteilung

Teilt man zu entwickelnde Anwendung in Schichten auf, so kommt man zu folgendem Ergebnis. Die von Tanenbaum (v. Tanenbaum, 2007) verwendete Datenbankschicht ist in dieser Anwendung äquivalent zum Informationdatenspeicher.

Aufgrund eingeschränkter Speicher-Ressourcen auf dem Mobiltelefon muss dort auf eine dauerhafte Datenhaltung verzichtet werden, das lässt sich durch die Auslagerung der Daten auf den Server bewerkstelligen. Um zusätzlich die gesamte Funktionalität eines Mobiltelefons nutzen zu können, also Beispielsweise die Kamera reicht es nicht aus, nur die Benutzerschnittstelle auf dem Client zu belassen. Es muss auch ein gewisser Teil Anwendungslogik auf dem Client zurückbleiben der diese Nutzung ermöglicht. Die zu erstellende Anwendung lässt sich in Tanenbaums Skala also in der Mitte einordnen (c).

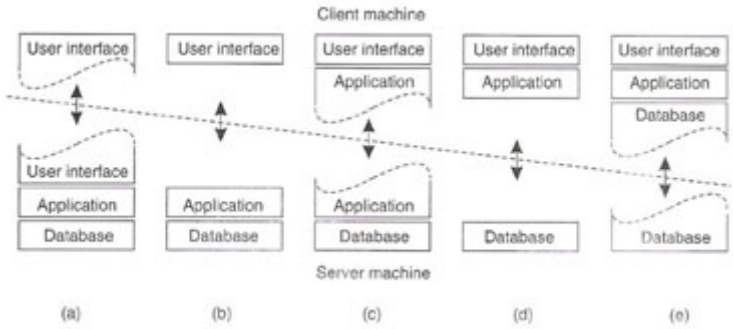


Abbildung 6.1: Verteilung nach Tanenbaum

6.2 Systemtopologie

Durch die bereits aufgestellten Einschränkungen hinsichtlich Kommunikation und Positionierung der Clients (Verwendung von Bluetooth), beschränkt sich die Anzahl möglicher Serversetups. Die aus der Hardwarewahl resultierende Verschlechterung der Reichweite und der Übertragungsleistung legt Nahe, eine Dezentrale verteilte Lösung anzustreben.

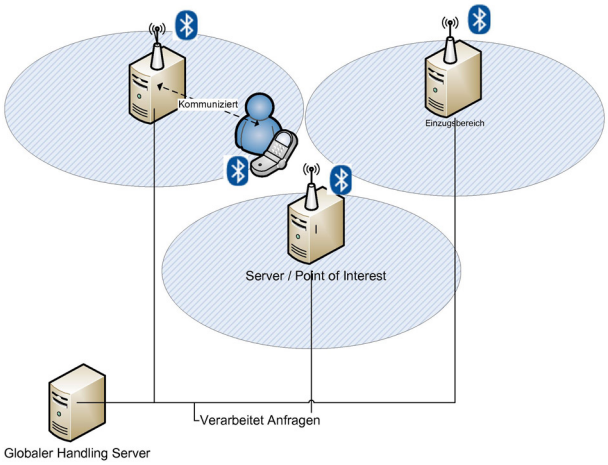


Abbildung 6.2: System-Topologie

Die Systemtopologie ergibt sich daher wie folgt:

Für jeden POI wird ein Serverrechner benötigt. Dieser ist zuständig, für das Erkennen von Clients in Reichweite, die Bekanntmachung bei diesen und wenn gewünscht für die Bereitstellung von Inhalten und Diensten. Dabei ist zu beachten, dass der Serverrechner durchaus nur als Zugangspunkt dienen kann. Die Inhalte bzw. die Dienste würden in diesem Fall von

einem weiteren Rechner bereitgestellt werden. Mit alternativen Kommunikations- und Positionierungsmethoden wären mehrere weitere Szenarien denkbar.

Auch wenn die Positionierung und die Inhalts-/Dienstbereitstellung in dem hier vorhandenen Fall auf einem Rechner konzentriert wurde, sollte im Designprozess zu Gunsten von Skalierbarkeit und Erweiterbarkeit auf eine lose Kopplung der Komponenten geachtet werden. Dies ermöglicht eine nachhaltige Anpassbarkeit an neue Gegebenheiten, wie zum Beispiel die Erhöhung von Reichweite und Sendeleistung, durch Nutzung von WLAN-Frequenzen, die im Bluetooth 3.0 Standard angestrebt wird (Rene Melzer, 2008).

6.3 Protokoll

Ein Protokoll in der Informatik ist eine Vorschriftensammlung bezüglich des Versands (Senden und Empfangen) von Nachrichten. Es legt, zusätzlich zu der Reihenfolge in der Nachrichten auftreten dürfen, den Inhalt und dessen Format fest.

6.3.1 Anforderungen

Im vorliegenden Fall muss zu spezifizierende Protokoll folgendes Leisten:

- Funktionsfähig auf Client und Server
- Nutzbar über Bluetooth
- Informationsaustausch ermöglichen
  - Zwischen Client und Server
    - Rendezvous-Nachrichten
    - Datenaustausch (Darstellbare Informationen, Clienteingaben)
  - Zwischen Server und Server
    - Anmeldung/Abmeldung
    - Client-Anfragen weiterleiten und beantworten

Aufgrund der fehlenden Reflektionsfunktionaltiät in der CLDC Implementierung ist der Einsatz einer *Middleware* nahezu Unmöglich. Es fehlt die Möglichkeit aus Metainformationen zur Laufzeit eine Klasse zu generieren. Somit muss auf vorfertigte Klassenstrukturen zurückgegriffen werden. Diese werden zur Laufzeit instanziert und mit den nötigen Informationen konfiguriert.

6.3.2 Daten

Struktur

Orientiert an dem Aufbau des HTTP-Protokolls wird auch das zu entwickelnde Protokoll Daten in Kopf- (Header) und Körperdaten (Body) aufteilen.

Kopfdaten

In den Kopfdaten müssen mindestens enthalten sein:

Messagetype – Also die Art der Anfrage: Request, Response, Announcement

Bodysize – Größe der Körperdaten in Byte.

Connectiontype – Welcher Verbindungstyp soll genutzt werden, synchron oder asynchron

Optionale Kopfdaten sind:

Target/Source – Anhand der Werte von Target und Source kann festgelegt werden woher die Nachricht kam und wer der vorgesehene Empfänger ist.

Mimetype – Der MIME-Typ legt fest wie die Anwendung die empfangenen Daten interpretieren soll.

Körperdaten

In den Bodydaten werden die jeweiligen Informationen übertragen.

Die Trennung von Meta- und Informations-Daten ermöglicht es, während der Übertragung, genauer nach der Übertragung der Kopfdaten, zu entscheiden, wie weiter verfahren werden soll. Anhand des Parameters Bodysize kann beispielsweise entschieden werden, wie mit den ankommenden Daten verfahren werden soll. Ist genug flüchtiger Speicher vorhanden kann die Datenlast ins RAM geschrieben werden. Ist dies nicht der Fall muss eine Cache-Datei erstellt werden um die Informationen aufzunehmen. Zudem wäre es möglich, vor der Übertragung von großen Dateien den Benutzer über den Zeitbedarf aufzuklären und ihm zu ermöglichen den Transfer abzubrechen.

Für die Strukturierung der Kopf- und Körperdaten wird XML zum Einsatz kommen. Da XML zusätzlich zu den Nutzdaten auch Meta-Daten enthält ist es einfacher zu verarbeiten als Daten bei denen allein durch deren Reihenfolge bestimmt wird, welche Bedeutung sie haben. Zur Formalisierung der Protokollstruktur kann eine Document Type Definition eingesetzt werden.

Format

Die Nutzdaten können verschiedenartig kodiert sein, binär, nur Text oder als Mischform. Abhängig von dem in den Kopfdaten festgelegten Typ müssen die Informationen unterschiedlich interpretiert werden.

6.3.3 Verbindungstyp

Um eine möglichst dynamische Kommunikation zu ermöglichen sind zwei Verbindungsarten erforderlich: Anfrage-Antwort und Direkte Verbindung.

Im Anfrage-Antwort-Modus stellt der Client eine Anfrage an den Server, das bedeutet er baut eine Stream-Verbindung auf sendet einmalig eine Anfrage. Der Server erkennt anhand der Kopfdaten, dass es sich um eine Einzelanfrage handelt. Dementsprechend liest er die Körperdaten bis zu der angegebenen Maximallänge und verarbeitet sie. Ist die Verarbeitung abgeschlossen, wird die Verbindung unterbrochen. Dieser Modus bietet sich für die Nutzung des Clientgerätes als Informationsanzeige an.

Der Ablauf einer Verbindung im Direkten-Verbindungsmodus gestaltet sich unterschiedlich. Nachdem die Kopfdaten der Client-Anfrage ausgelesen wurden, weiß der Server, dass es sich um eine Direktverbindung handelt. Das heißt, es gibt für die Körperdaten keine Maximallänge. Der Datenstrom wird erst unterbrochen, wenn der Server eine bestimmte Terminalzeichenkette vom Client empfangen hat. Die Verarbeitung beziehungsweise Weiterleitung dieses Datenstromes kann demnach zeitgleich mit dem Empfang der Daten geschehen. Bei diesem Verbindungsmodus wird ein sonst bei hoher Anfragefrequenz entstehender Overhead an Kopfdaten vermieden.

Im Gegensatz zum Anfrage-Antwort-Modus, kommt dieser Verbindungstyp vor allem Anwendungen zu gute, die das angeschlossene Clientgerät als direktes Ein-/Ausgabegerät nutzen zum Beispiel die Fernsteuerung von Modelautos oder Ähnlichem. Da es sich um eine bidirektionale Kommunikation handelt, wird die Clientseite ebenfalls in den Direktverbindungsmodus gebracht. Das ermöglicht es dem Server, bzw. der die Direkte-Anfrage verarbeitenden Instanz, direkt Daten an den Client zu übermitteln. Der Inhalt dieser Daten sowie die Reaktion der Clientanwendung, bleiben der jeweilig verarbeitenden Instanz überlassen. Zu Testzwecken wurde bisher nur die Ausgabe einer Textmeldung nach vorherigem löschen des Bildschirms auf dem Client implementiert. Denkbar wäre eine Implementierung mit Zeichenbefehlen, die dem Client mitteilen was darzustellen ist.

6.3.4 Umsetzbarkeit des Protokolls mit Bluetooth

Wie bereits im Bluetooth-Kapitel 3.6 in den Grundlagen beschrieben, bietet Bluetooth mit dem Serial Port Profile einen Bidirektionale Kommunikationsmechanismus. Dieser wird mit dem im JSR-82 5.1.3 spezifizierten Paket über das GCF 5.1.3 auch zugänglich. Über diese serielle Verbindung lassen sich die hier erwähnten Daten ohne Einschränkungen übertragen.

6.4 Informationsformat

Dieser Abschnitt beschreibt das Format in dem Informationen zwischen Client und Server ausgetauscht werden können.

6.4.1 Anforderung

Wie bei der Konzeption der gesamten Clientanwendung, muss auch bei der Entwicklung eines Formats zur Repräsentation und Darstellung von Informationen auf dem Mobiltelefon auf die gleichen Einschränkungen Rücksicht genommen werden.

Displaygröße – Im Gegensatz zu Desktoprechnern oder Notebooks ist der Anzeigebereich eines Mobiltelefons nur einen Bruchteil so groß. Das wirkt sich insbesondere auf die Menge der darstellbaren Information aus.

Speichergröße – Begründet mit der geringen Größe des flüchtigen Speichers können großen Datenmengen nicht ständig zur Verfügung gehalten werden.

6.4.2 Format

Um ein möglichst variables System für den Austausch von Informationen zu erhalten wird XML zum Einsatz kommen. Als sogenannte Markup-Language ermöglicht es XML, Metainformationen, also Informationen die das Rohdatum näher beschreiben, zu übertragen.

Mithilfe dieser Metainformationen lassen sich die Daten einfacher interpretieren und verarbeiten. Da XML nicht wie HTML einen festgesetzten Vorrat an erlaubten Tags und Elementen besitzt, sondern dieser vom Einsatzzweck abhängig gemacht werden kann (vgl. DTD), erfüllt es die Forderung nach Erweiterbarkeit hinsichtlich neuer Technologien in vollem Umfang.

6.4.3 Besonderheiten

Der Anzeigebereich von Mobiltelefonen reicht nur für eine geringe Menge an Information aus, dass bedeutet wiederum, dass ein Informationsangebot welches viele Daten umfasst ständig Verbindungen auf und abbauen müsste, um diese zu laden. Um diesem Problem Abhilfe zu leisten wird es ermöglicht, ähnlich dem WML-Cards Konzept, mehrere Inhalte in einem Datendokument zu übertragen. Diese Sektionen sind durch das <document>-Tag gekennzeichnet. Die Navigation zwischen diesen Sektionen und weiteren nicht eingebetteten Ressourcen erfolgt über Navigationsschaltflächen („<button type="navigate>").

Binärdaten

Eine weitere Besonderheit ist die Möglichkeit Binärdaten im Datendokument übertragen zu können. Dies ermöglicht es, zum Beispiel Bilder direkt im Dokument einzubetten, so dass kein weiterer Verbindungsaufbau benötigt wird. Da die Daten vom als XML verarbeitet werden und Binärdaten interpretiert als Text teils XML-Sonderzeichen enthalten können, müssen diese Codiert werden. Ein Format das sich hierfür anbietet ist das Base64-Format, da die für die Kodierung benötigten Klassen auf Client und Server bereits verfügbar sind

Die möglichen Inhalte können der Document Type Definition entnommen werden A.1.

6.5 Server

Die Serverkomponente lokalisiert die Clients und stellt ihnen zu ihrer jeweiligen Position Information und Dienste zur Verfügung.

6.5.1 Anforderungen

Zugriff auf Positionsdienst

Im zu entwickelnden Fall, in dem jeder POI mit einem Server ausgestattet wird, ist die Aufgabe des Positionsdienstes relativ simpel. Der Dienst prüft in einem bestimmten Intervall ob sich kompatible Client-Geräte in Reichweite befinden und meldet diese. Analysiert man den Zugriff allerdings unter dem Gesichtspunkt, dass die Positionierung nicht lokal sondern durch einen zentralen Server durchgeführt wird, wird deutlich das die Komponente explizit implementiert werden muss.

Auswertung der Positionsinformationen

Ähnlich dem Zugriff auf einen Positionsdienst, gestaltet sich auch die Auswertung der Positionsinformationen in der aktuellen Konfiguration als sehr einfach. Die Komponente muss jeden vom Positionsdienst gemeldeten Client als gültig und in Reichweite ansehen. Dies ist wiederum durch die dezentrale Annäherungsbasierte Ortung begründet. Auch hier wird der Sinn der Komponente erst durch den Hinweis auf eine mögliche dezentrale Ortungsstruktur deutlich. In diesem Fall müsste ausgewertet werden, ob die erhaltenen Client-Positionen im Einzugsbereichs des aktuellen Servers liegen und dementsprechend reagiert werden.

Kontaktaufnahme/Kommunikation mit den Clients

Um das Push-Dienst-Paradigma zu erfüllen, muss der Server mit einem oder mehreren Clients Kontakt aufnehmen können. Dies geschieht hauptsächlich zur Bekanntmachung, kann aber in Einzelfällen auch zur proaktiven Informationsübermittlung genutzt werden (Informations-Ticker). Dabei soll das bereits unter 6.3 spezifizierte Protokoll Anwendung finden.

Verwaltung von weiteren Servern

Um andere Server von den aktuell verbundenen Clients unterrichten zu können soll es möglich sein, weitere Server zu registrieren und mit ihnen Kontakt aufnehmen zu können.

Verarbeitung von Clientanfragen

Die Hauptaufgabe eines Servers ist es, dem Client Informationen und Dienste zur Verfügung zu stellen. Dabei soll es möglich sein, dass die Beantwortung der Anfrage auch nicht vom angefragten Server, sondern von einem entfernten Rechner geschehen kann. Die Verarbeitung soll also Ortstransparent erfolgen können.

6.5.2 Strukturierung

Anhand der Verantwortlichkeiten lässt sich die Anwendung somit in verschiedene Module Unterteilen:

Steuerung (Control)

Das Steuerungsmodul enthält die Klassen und Funktionen die für den Betrieb, also Start- und Stop des Servers, sowie dessen Konfiguration zuständig sind.

Positionierung (Positioning)

Beinhaltet Funktionalität für die Positionierung, Abstandsmessung und deren Umrechungen.

Location Service

Spezifiziert die Schnittstelle für Nutzung eines Location Services.

Nachrichten Verarbeitung (Handling)

Geschäftslogik für die Verwaltung von Anfrageverarbeitern, weiteren Servern und des Client-Rendevous-Prozesses.

Kommunikation

Im Kommunikationsmodul befindet sich die Logik für den Nachrichtenaustausch. Dieses Modul wird sowohl vom Server wie auch vom Client genutzt.

6.5.3 Schichtung

Zur weiteren Strukturierung der Server-Anwendung werden die oben genannten Module in einem Schichtenmodell angeordnet.

6.5.4 Abgrenzung

Aufgrund der geringen Zeitvorgabe für diese Arbeit wird die Verarbeitung von Client-Anfragen auf den verbundenen Server-Rechner beschränkt. Eine Verteilung der Bearbeitungslogik würde einen stark erhöhten Zeitaufwand bedeuten. Gleiches gilt für den Einsatz von Globalen Informationsservern zur Nutzung von Ticker-Diensten. Hierfür müsste ebenfalls eine verteilte Struktur implementiert werden, die den Rahmen dieser Arbeit übersteigt.

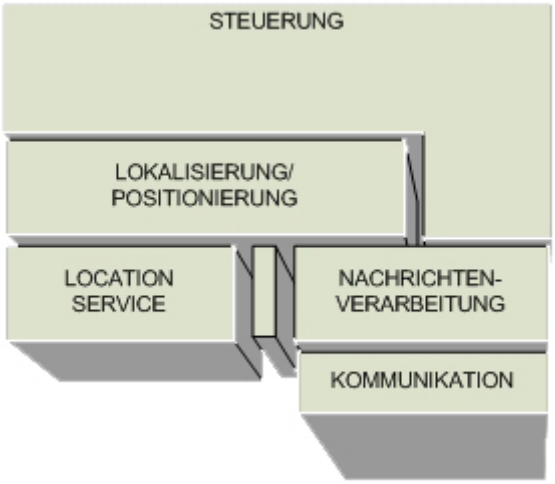


Abbildung 6.3: Schichtung der Serverkomponenten

6.5.5 Komponenten

Im Folgenden werden kurz die Hauptkomponenten der Serveranwendung erläutert.

Positioning:

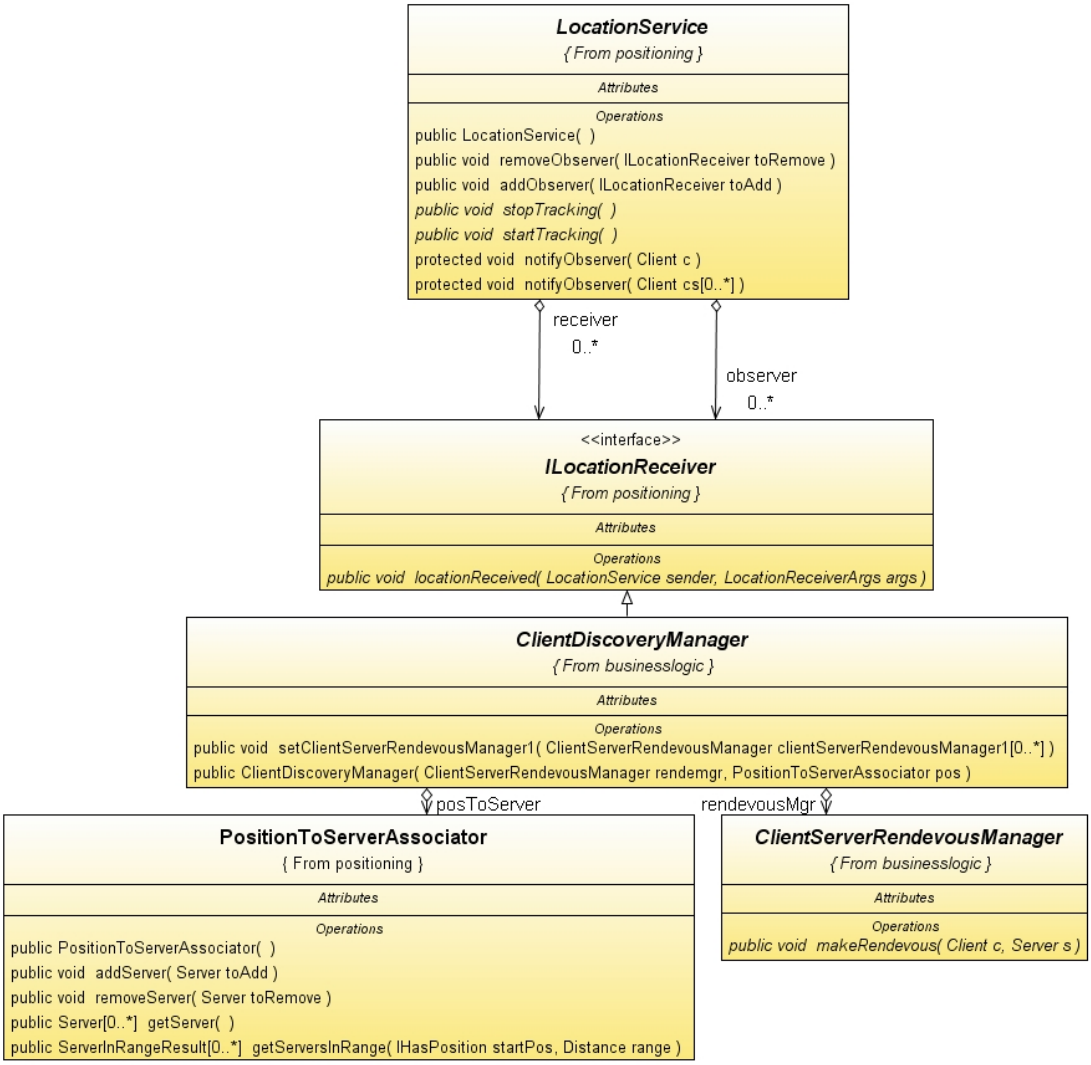


Abbildung 6.4: UML Klassendiagramm Server: Positioning

Position – Basisklasse für Positionsdaten.

LocationService – Abstrakte Basisklasse. Stellt Grundfunktionen für die Lokalisierung von Clients zur Verfügung. Einzige konkrete Implementierung ist die BluetoothLocationService-Unterklasse.

PositionToServerAssociator –Basisklasse für die Zuordnung von einer Client-Position zu einem oder mehreren Servern. In der Basisklasse sind rudimentäre Funktionen für die Verwaltung der vorhandenen Server und für die Berechnung von Abständen definiert.

Geschäftslogik:

ClientServerRendevousManager – Abstrakte Basisklasse für die Bekanntmachung vom Server beim Client. Eine konkrete Implementierung ist die Klasse BluetoothRendevousManager.

ClientDiscoveryManager – Abstrakte Klasse. Kapselt das Rendevousmanager-Objekt und den PositionToServerAssociator (PTSA). Registriert sich beim PTSA für den Empfang von Zuordnungsinformationen und leitet diese an den Rendevousmanager weiter. Konkrete Implementierung ist der BluetoothClientDiscoveryManager.

Handling:

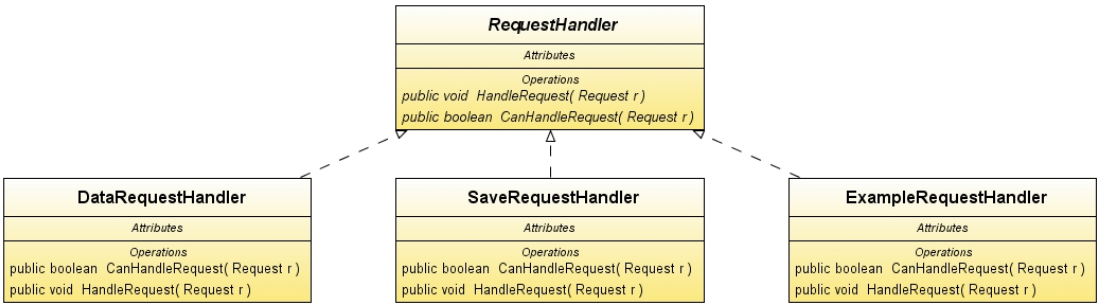


Abbildung 6.5: UML Klassendiagramm Server: Handling

RequestHandler – Abstrakte Basisklasse für die Beantwortung von Client-Anfragen. Konkrete Implementierung ist der DataRequestHandler (DRH). Der DRH stellt dem Client Daten aus dem Serverdateisystem zur Verfügung.

SaveRequestHandler – Konkreter RequestHandler. Speichert übermittelte Binärdaten und gibt Erfolgs- bzw. Fehlermeldung zurück.

DataRequestHandler – Konkreter RequestHandler. Liest Datei aus Server-Dateisystem und übermittelt sie an den Client. Vergleichbar mit einem HTML-Server.

ExampleRequestHandler – Konkreter RequestHandler. Zu Testzwecken erstellter RequestHandler. Nur Logging Funktionalität.

Request

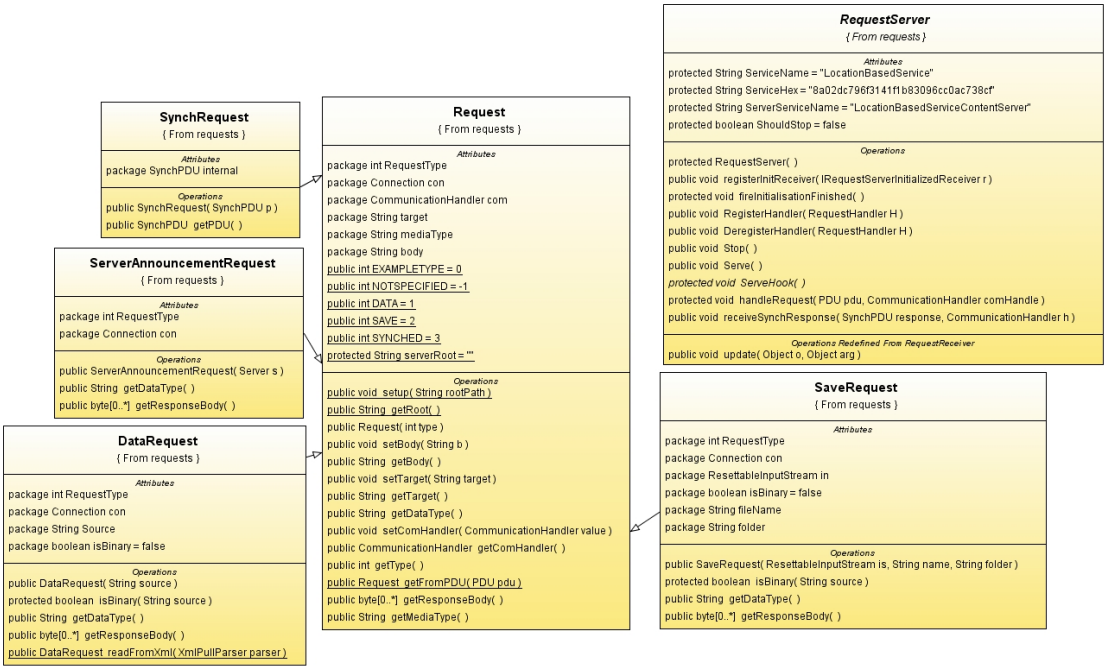


Abbildung 6.6: UML Klassendiagramm Server: Request

RequestServer – Abstrakte Klasse. Implementiert Steuerungsfunktionen für Start und Stop. Implementierungen stellen den Hauptzugangspunkt von Client-Anfragen dar, sie warten auf eine Verbindung und delegieren diese dann an den CommunicationHandler. Verwaltet die vorhandenen RequestHandler und delegiert ankommende Requests an Sie. Konkretisierung ist der BluetoothRequestServer.

Request – Basisklasse für Clientanfragen. Werden aus PDUs erstellt und an einen Request-Server zur verarbeitung weitergeleitet. Unterklassen sind zum Beispiel die DataRequest-Klasse oder die ServerAnnouncementRequest-Klasse, die bei der Bekanntmachung von Server bei Client zum Einsatz kommt.

ServerAnnouncementRequest – Konkrete Unterklasse von Request. Repräsentiert eineServer-Rendevous-Nachricht.

DataRequest – Konkrete Unterklasse von Request. Repräsentiert eine Anfrage nach einer Datei.

SaveRequest – Konkrete Unterklasse von Request. Repräsentiert eine Speicher-Anfrage.

Communication

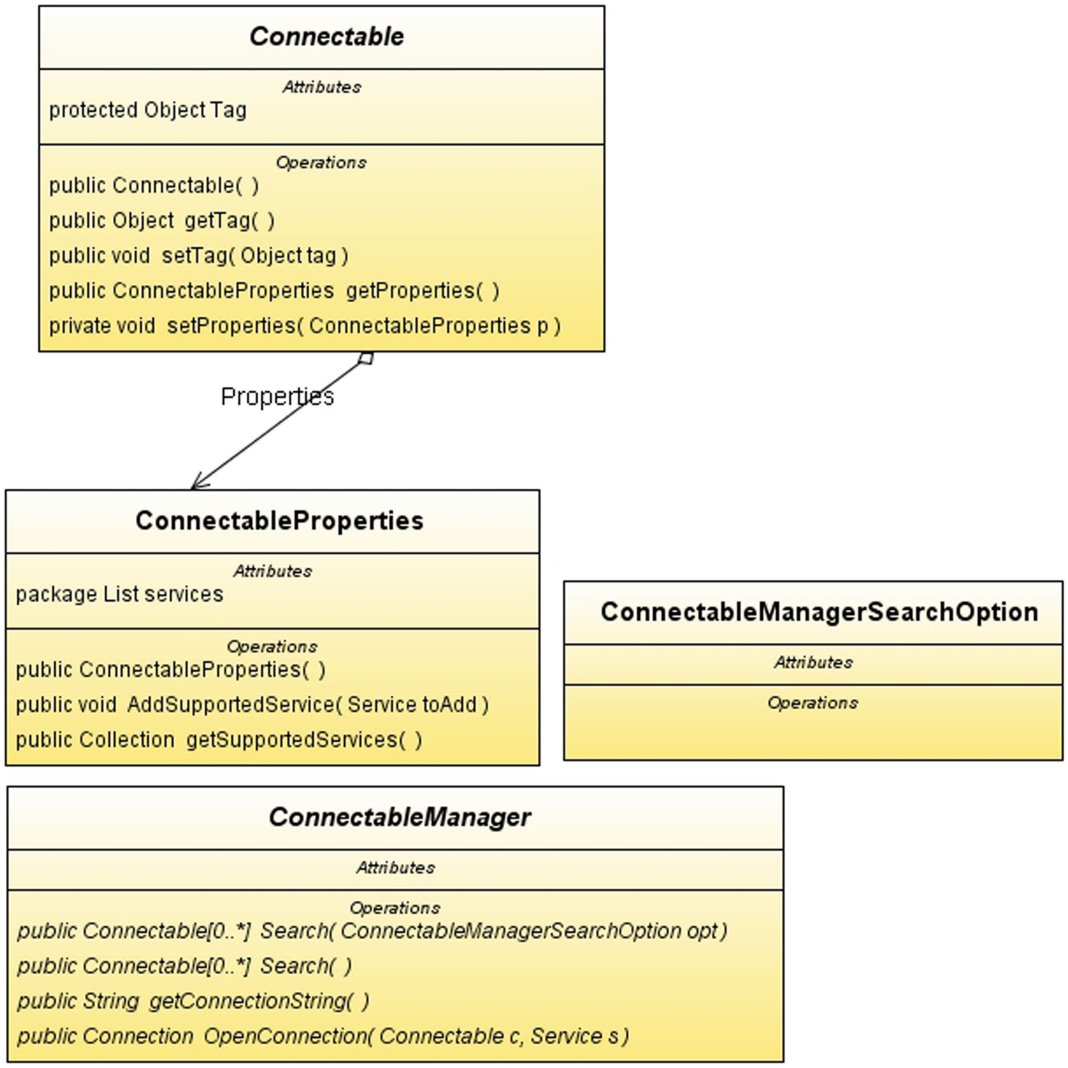


Abbildung 6.7: UML Klassendiagramm Server: Communication

Connectable – Abstrakte Basisklasse. Repräsentiert ein physikalisches Gerät, zu dem eine Verbindung aufgebaut werden kann. Konkrete Unterklasse ist die BluetoothConnectable-Klasse.

ConnectableManager – Abstrakte Basisklasse. Stellt Funktionalität bereit, um Connectables zu suchen und eine Verbindung zu Ihnen aufzubauen. Konkreter Unterklasse ist der BluetoothConnectableManager.

ConnectableManagerSearchOption – Optionen die für die Suche nach verfügbaren Connectables notwendig sind.

ConnectableProperties – Repräsentiert Eigenschaften von Connectables. Anhand dieser Eigenschaften kann geprüft werden, ob das Connectable kompatibel zu der Anwendung ist. Im konkreten Bluetooth Fall wird hiermit vermerkt, welche Bluetooth-Dienste das Gerät unterstützt.

6.6 Client

Die Client ist der Teil der verteilten Anwendung, mit der der Benutzer auf seinem Mobiltelefon interagiert.

6.6.1 Anforderungen

Bereitstellung von Position

Um ortsbasierte Dienstleistungen anbieten zu können, muss der Server die Position des Clients kennen. Da das Clientgerät von sich aus nicht ortbar ist, abgesehen von der Positionierung durch den Mobilfunkprovider, muss es den Server aktiv unterstützen. Im vorliegenden Fall, bei Verwendung von Bluetooth, tut es dies dadurch, dass es einen Dienst per SDP registriert und eine Serververbindung öffnet. Sollte später eine weitere Konfiguration eingesetzt werden, mit einer zentralen Positionierungslösung zum Beispiel GPS, könnte der Client in bestimmten Zeitabständen seine Position an einen oder mehrere Server übermitteln.

Kontaktaufnahme/Kommunikation zu/mit dem Server

Nachdem ein Server erkannt, beziehungsweise aktiv bekanntgemacht wurde hat der Anwender die Möglichkeit die angebotenen Dienste zu nutzen. Anhand einer durch das spezifizierte Protokoll übertragenen Anforderung, wird dem Server dies mitgeteilt. Diese Kommunikation kann sowohl im Anfrage-Antwort-Verfahren sowie auch im Direkt-Verfahren statt finden.

Darstellung von Information

Der Client muss in der Lage sein, das im Kapitel 6.4 festgelegte Format interpretieren und darstellen zu können.

Konfiguration

Die Client-Software muss konfigurierbar sein. Das bedeutet zum Beispiel, dass die Art der Benachrichtigung für den Anwender, eines in Reichweite gekommenen Servers durch den Benutzer variierbar ist. Denkbar wären hier die Möglichkeiten: Audiobenachrichtigung, ein Ton, wenn vorhanden taktil durch Vibration oder Visuell durch eine Nachricht im Display.

Erweiterbarkeit für Gerätespezifische Eigenheiten

Gerätespezifische Eigenheiten müssen für die Client-Software gemacht werden.

Sonstiges

Wie zuvor erwähnt, muss bei Mobiltelefonen besonders auf die Hauptspeichernutzung geachtet werden. Der geringe vorhandene Speicher erfordert es große Datenmengen auszulagern.

6.6.2 Strukturierung

Anhand dieser Anforderungen lassen sich folgende Komponenten bilden:

GUI (Graphical User Interface)

Die GUI-Komponenten sind für die Darstellung der Informationen und der Steueroberfläche zuständig.

Positionierung (Positioning)

Im Modul Positioning befindet sich die Logik für die Serverunterstützung bei der Positionsfindung des Clients.

Nachrichtenverarbeitung (Handling)

Diese Komponente ist zuständig für die Verarbeitung von Nachrichten, d.h. sowohl die Erstellung und ihr Versand als auch der Empfang und Verarbeitung von allen Anfragen und Antworten fallen in ihren Aufgabenbereich.

Kommunikation

Im Kommunikationsmodul befindet sich die Logik für den Nachrichtenaustausch. Dieses Modul wird sowohl vom Server wie auch vom Client genutzt.

Caching

Um einer Füllung des Hauptspeichers vorzubeugen, wird ein Teil des Datenverkehrs im nicht flüchtigen Speicher zwischengespeichert. Diese Funktionalität wird vom Caching-Modul bereitgestellt.

PlugIn

Für die Bereitstellung von gerätespezifischer Funktionalität, wird ein PlugIn-System implementiert. Aufbauend auf einer Abstrakten Klassenstruktur können damit je nach Konkretisierung die unterschiedlichen Eigenheiten von verschiedenen Mobiltelefonen genutzt werden.

6.6.3 Schichtung

Wie der Server können auch die Client Komponenten Schichten angeordnet werden. Dies dient der Strukturierung und der Übersichtlichkeit.

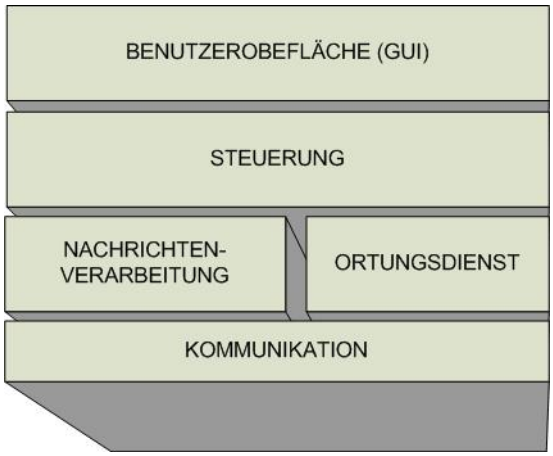


Abbildung 6.8: Strukturierung des Clients

6.6.4 Komponenten

Die Hauptbestandteile des Clients sind aufgeteilt nach ihrer Funktion:

Handling

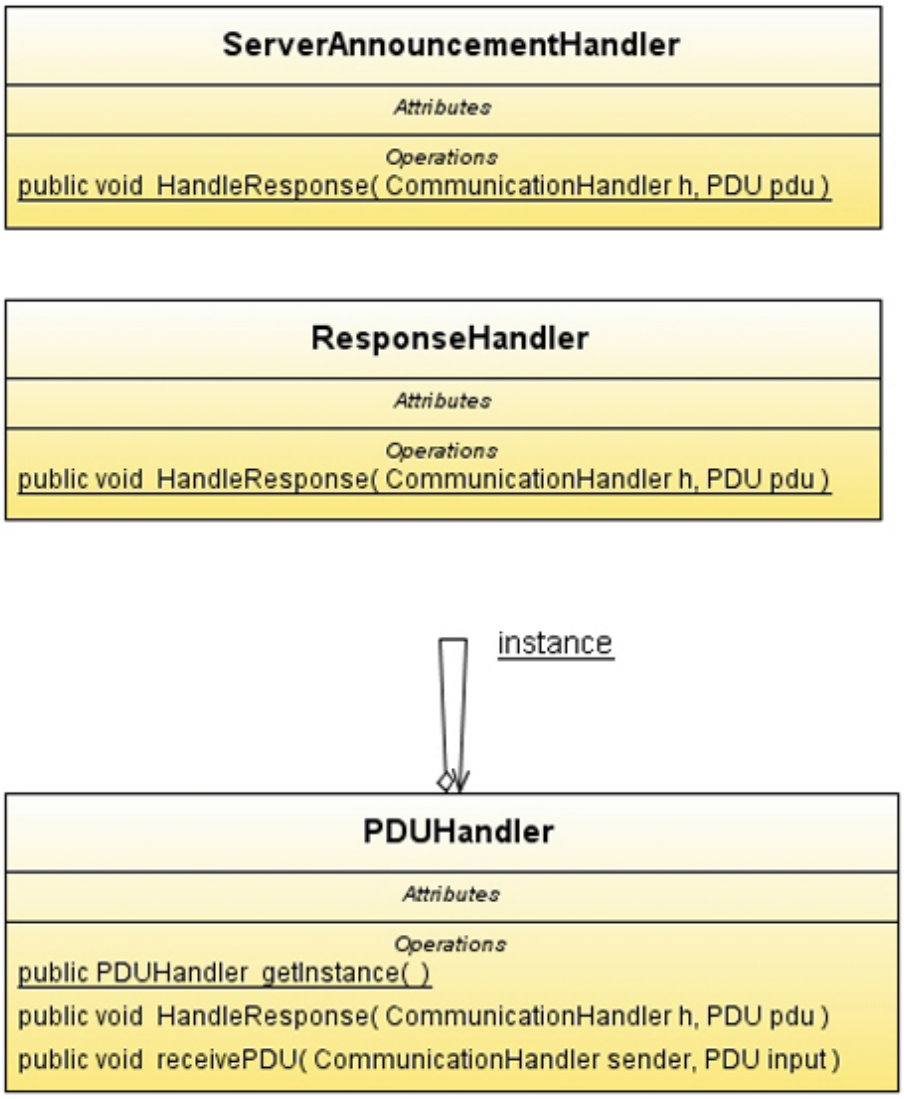


Abbildung 6.9: UML Klassendiagramm Client: Handling

PDUHandler – Singleton. Stellt Methoden für die Delegierung der ankommenden Nachrichten an den jeweiligen Verarbeiter bereit.

ServerAnnouncementHandler – Verarbeitet Server-Rendevous-Nachrichten.

ResponseHandler – Verarbeitet alle nicht Server-Rendevous-Nachrichten.

Management

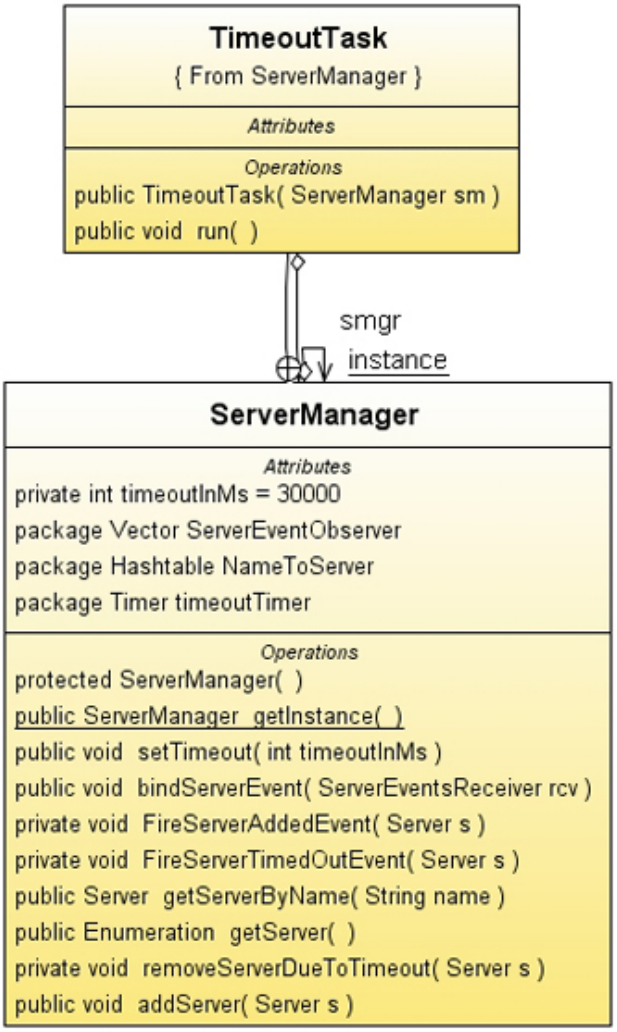


Abbildung 6.10: UML Klassendiagramm Client: Servermanagement

ServerManager – Singleton. Verwaltet die bekannten Server. Führt eine Liste mit Erstellungszeitpunkten mit und entfernt Server, die sich nicht in einem bestimmten Intervall erneut melden. Dies vermeidet den Zugriff auf Server die sich außer Reichweite befinden.

TimeoutTask – Runnable. Signalisiert dem ServerManager die vorhandenen Server auf Gültigkeit zu überprüfen.

UI

Enthält verschiedene Klassen die die Oberfläche der Anwendung repräsentieren. Diese enthalten Logik zur Darstellung der Menüs und Dialoge auf dem Mobiltelefon Display.

ScreenManager – Zentrale Singleton-Klasse zur Darstellung und Verwaltung der verschiedenen Dialoge.

Renderscreen – Verarbeitet das Informationsdokument und stellt das Ergebnis dar.

SplashScreen – Titelbildschirm der am Anfang der Anwendung dargestellt wird.

ServerSelectionScreen – Übersicht über die in sich in Reichweite befindenden Server und Zugriff auf Ihre Inhalte.

MainMenu – Das Hauptmenü der Client-Anwendung. Zugriff auf Optionen und den ServerManager.

6.6.5 Weitere Komponenten

Bestandteile des Clients die zur besseren Strukturierung ausgelagert wurden.

Rendering

Umsetzung der Informationsdokumentenstruktur in Midlet-Steuerelemente

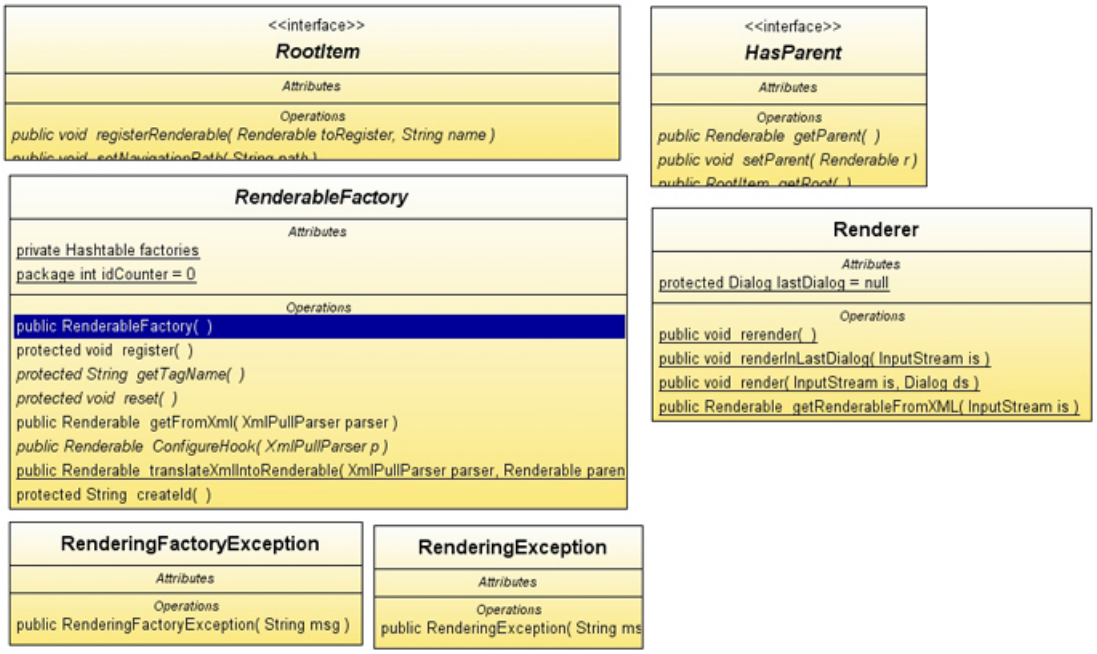


Abbildung 6.11: UML Klassendiagramm Client: Rendering

Renderer – Singleton Klasse die den zentralen Zugriffspunkt für die Übersetzung von XML in Steuerelemente darstellt.

RenderableFactory – Abstrakte Basisklasse für Fabrik-Klassen für die Umsetzung von XML-Tags in Steuerelemente. Konkrete Implementierungen sind zum Beispiel: Document-Factory, ImageFactory oder die InputButtonFactory. Alle Implementierungen werden in einer Abbildung XML-Elementname zu Fabrik gespeichert um später einfacheren Zugriff zu haben.

RootItem – Klassen die dieses Interface implementieren sind Wurzel-Elemente in der Darstellungsbaumstruktur.

HasParent – Klasse die dieses Interface implementieren können weitere Unterelemente beinnhalten.

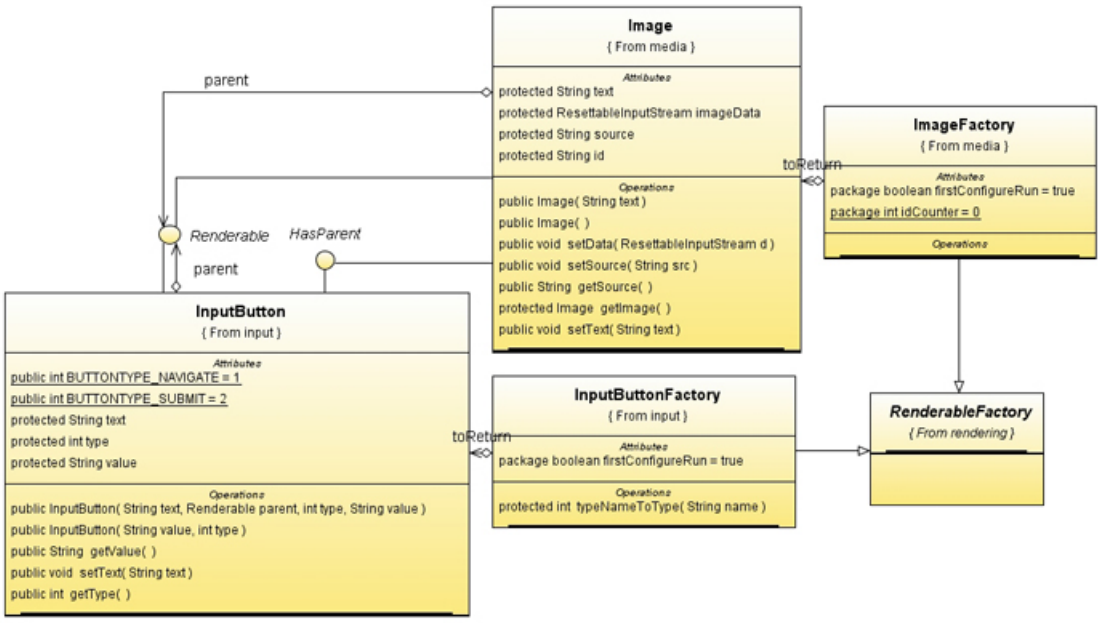


Abbildung 6.12: UML Klassendiagramm Client: Renderables

Image – Stellt ein darzustellendes Bild dar. Kann entweder direkt über Base64-Codierte-Binärdaten im Dokument gefüllt werden, oder mit einer Referenz auf das Serverdateisystem. Im zweiten Fall wird bei der Darstellung des Bildes eine weitere Anfrage an den Server gestellt.

InputButton – Repräsentiert eine Schaltfläche. Aktuell dient diese zur Navigation zwischen einzelnen Dokumenten und Unterdokumenten.

ImageFactory, InputButtonFactory – Fabriken für die Erstellung und Konfiguration der jeweiligen Elemente.

PlugIn

Repräsentation Handyspezifischer Funktionen. Implementiert verschiedene Tagging-Interfaces für die Typisierung hinsichtlich der Art des Rückgabewertes des PlugIns.

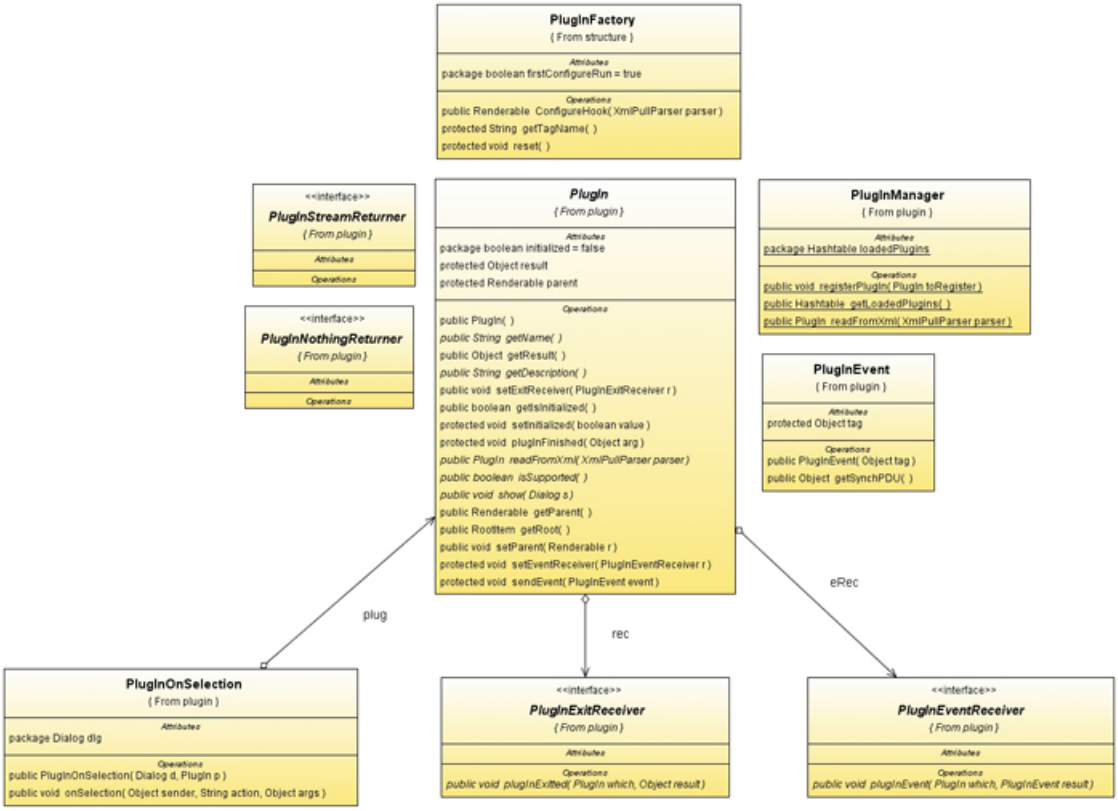


Abbildung 6.13: UML Klassendiagramm Client: Plugin

PlugIn – Abstrakte Basisklasse. Stellt Grundfunktionalität für Darstellung, Identifizierung und Aktivierung bereit.

PlugInOnSelection – Klasse die Aktionen implementiert, wenn das PlugIn aktiviert wird.

PlugInEvent – Zur Übermittelung von Nachrichten und Veränderungen die während der Ausführung des PlugIns stattfinden.

PlugInManager – Verwaltet und stellt zentralen Zugriffspunkt für die registrierten PlugIns zur Verfügung.

PlugInStreamReturner, PlugInNothingReturner – Tagging Interfaces die bestimmen wie bei Beendigung des PlugIns verfahren werden soll.

PlugInFactory – Fabrik-Klasse zur Erstellung Konfiguration eines PlugIns.

PlugInExitReceiver, PlugInEventReceiver – Interfaces für die Implementierung von Nachrichtenübermittelung.

6.7 Abläufe

Die wichtigsten Abläufe bei der Kommunikation und der Informationsverarbeitung werden im Folgenden Abschnitt beschrieben und in UML-Sequenzdiagrammen dargestellt. Diese Beschreibungen bilden Leitfäden für die spätere Implementierung.

6.7.1 Initiale Client-Server-Kommunikation

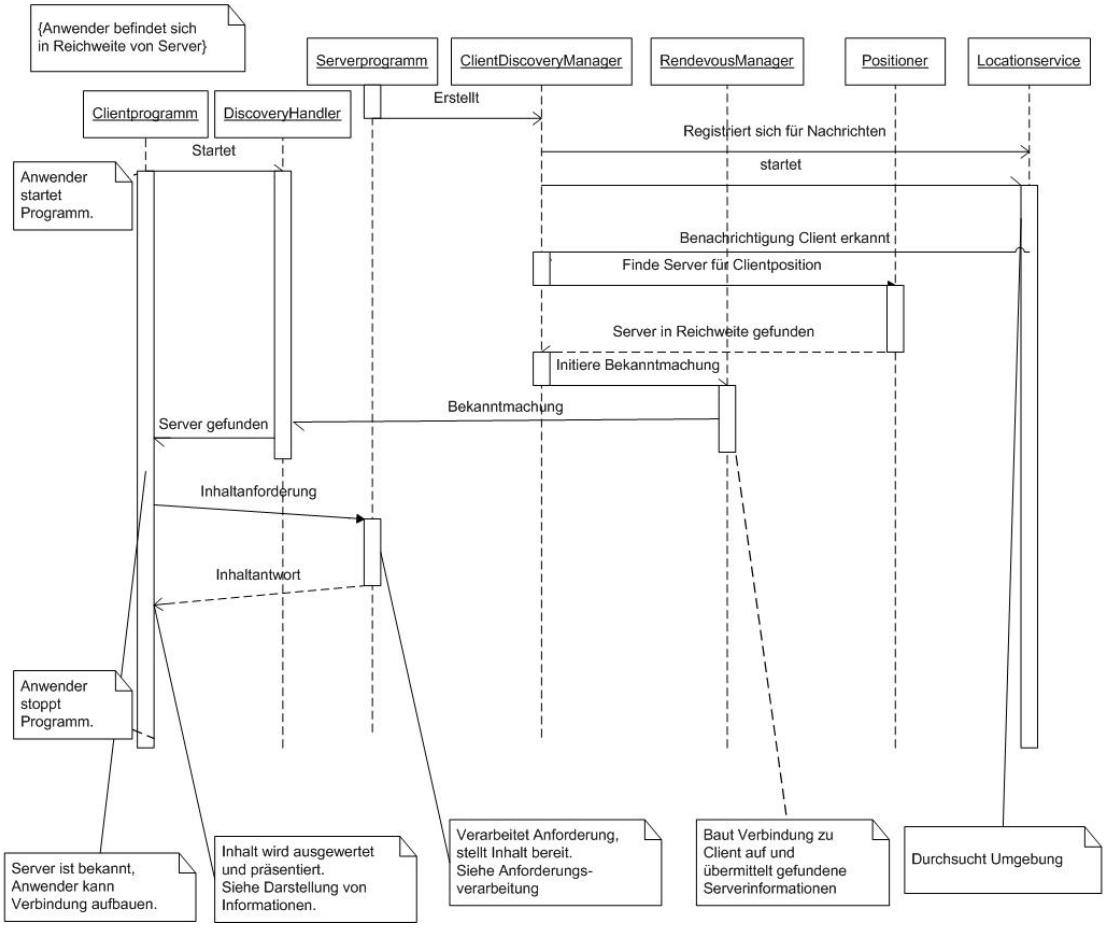


Abbildung 6.14: UML-Sequenzdiagramm Initiale Kommunikation

Ein Anwender betritt mit seinem Mobiltelefon, nachdem er die Clientanwendung gestartet hat, den Einzugsbereich eines bereits laufenden Servers. Der Server hat zuvor seine Lokalisierungs-Objekte initialisiert und gestartet (ClientDiscoveryManager). . Im vorliegenden Fall ist der Location Service ein lokaler Prozess der in einem bestimmten Zeitintervall nach Bluetooth-Geräten mit installiertem Client-Dienst scannt und diese meldet.

Sobald der Location Service die Präsenz des Clients erkennt, sendet er eine Nachricht mit der Position und einer eindeutigen Identifikation, Bluetooth Verbindungsstring, an alle Beobachter (ClientDiscoveryManager). Anhand der übermittelten Daten versucht der ClientDiscoveryManager einen Server zu finden, der die erkannte Position abdeckt, dafür ruft er den Positioner auf. Im vorliegenden Fall wird dieser Schritt übersprungen, da ein Annäherungsdetektionsverfahren genutzt wurde um den Client zu positionieren. Der Positioner sucht, in seiner vorab gefüllten Serverdatenbank, nach passenden Servern und gibt diese an den Anfragenden ClientDiscoveryManager zurück. Sollten passende Server entdeckt worden sein, wird der Rendevous-Vorgang initialisiert. Bei diesem Vorgang versucht der Rendevous Manager zum Client Kontakt aufzubauen und die Serverdaten zu übermitteln. Im konkreten Fall wird versucht eine Bluetooth-SPP-Verbindung zu öffnen und dann eine ServerAnnouncementMessage zu übertragen. Auf der Clientseite wartet seit Beginn der Anwendung der DiscoveryHandler darauf, dass sich ein Server verbindet. Dies geschieht mit einer Bluetooth Server Verbindung. In Konfigurationen mit nicht lokal eingeschränkter Postionierung (GPS) könnte er dazu genutzt werden, in bestimmten Zeitabständen automatisch die Position des Clients zu übermitteln und anhand dessen in Reichweite befindliche Server zu erfragen. Die empfangenen Daten werden vom Client interpretiert und in der Serverliste dargestllt. Der Client bekommt damit über ein Auswahlmenü die Möglichkeit, Informationen vom Server abzufragen. Tut er dies, wird mit Hilfe der bei der Bekanntmachung übertragenen Kontaktdaten, eine Verbindung zum Server aufgebaut und die Standard-Indexdatei angefordert 6.7.2. Der Server überträgt diese schnellstmöglich zum Client, der Sie empfängt und darstellt 6.7.3.

6.7.2 Server: Anforderungsverarbeitung

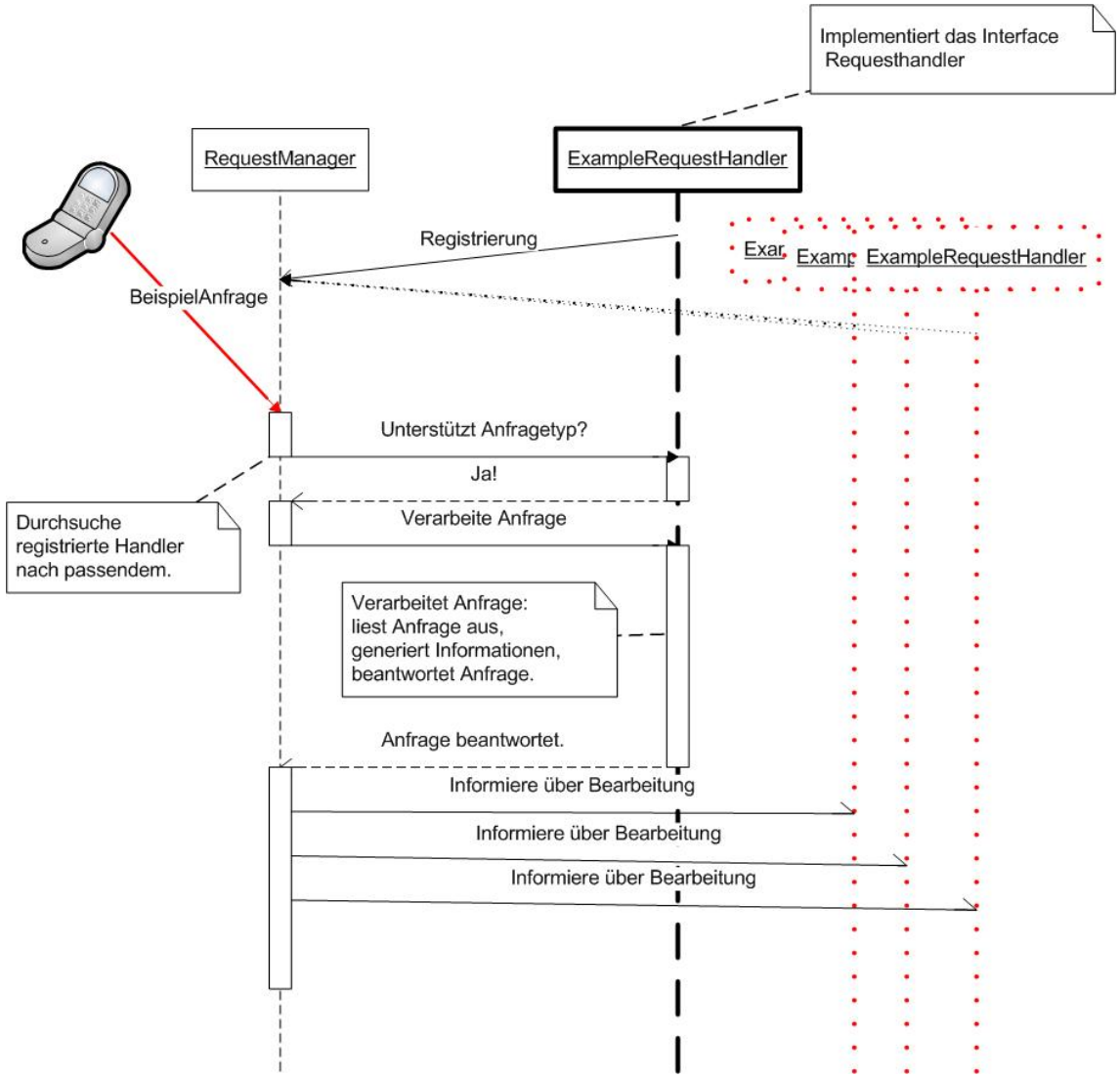


Abbildung 6.15: UML-Sequenzdiagramm Server Anforderungsverarbeitung Kommunikation

Bei Start der Serveranwendung werden die jeweiligen Anfrageverarbeiter im RequestManager registriert. Trifft nun eine Anfrage ein wird vom RequestManager geprüft welcher registrierte Verarbeiter den Typ der Anfrage unterstützt (Sollte kein passender Handler gefunden werden wird eine Standardnachricht an den Client gesendet). Der erste Verarbeiter der den angefragten Typ unterstützt bekommt den Auftrag die Anfrage zu verarbeiten. .Der RequestHandler bekommt dabei Zugriff auf das CommunicationHandler-Objekt, von dem die Anfrage empfangen wurde. Er hat somit die Möglichkeit direkt Daten an den Client zu übermitteln und die Verbindung wenn gewünscht zu unterbrechen.

Auf welche Art und Weise dies geschieht, bleibt der Implementierung des Verarbeiters vorbehalten. Nach Durchführung wird dies dem RequestManager signalisiert. Anhand der Rückgabe entscheidet der RequestManager ob die Behandlung endgültig ist, oder ob noch weitere Verarbeiter beauftragt werden sollen. Ist dies nicht der Fall werden die restlichen Verarbeiter trotzdem über die Anfrage informiert, bekommen aber nicht Möglichkeit auf sie zu antworten

6.7.3 Client: Informationsdarstellung

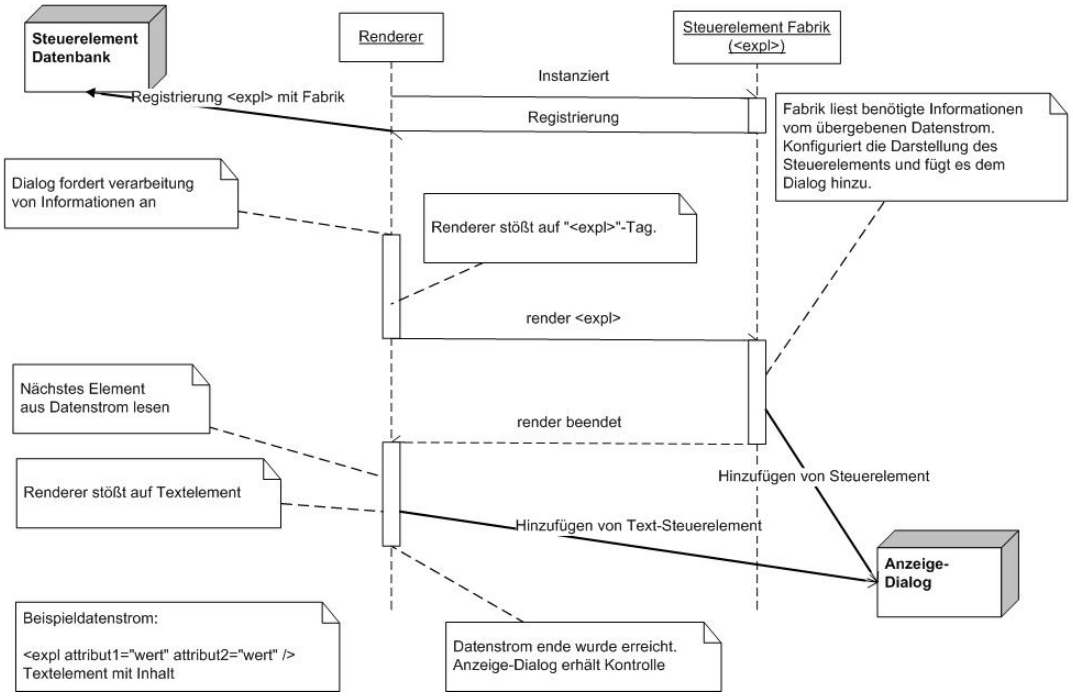


Abbildung 6.16: UML-Sequenzdiagramm Darstellung von Informationen

Nachdem das Renderer-Singleton die darzustellenden Informationen erhalten hat, werden diese interpretiert. Anhand des Elementnamens erkennt die Renderroutine, welche Art von Bedienelement für die grafische Repräsentation verwendet werden soll. Der Name ist ein Schlüssel in einer Abbildung, Elementname zu Steuerelement Fabrik. Die Fabriken wurden in vorher in einer Initalisierungsmethode instanziert dabei wird vom jeweiligen Konstruktor die Registrierung in der oben genannten Abbildung veranlasst. Der Renderer delegiert die Darstellungsaufgabe an das jeweilige Fabrikobjekt. Dieses liest auf dem übergebenen Datenstrom benötigte Attribute und wenn vorhanden Unterstrukturen aus und konfiguriert anhand dessen die Darstellung. Dabei wird aus den vom MIDP angebotenen Steuerelementen eine oder mehrere passende Repräsentationen gewählt. Eine Ausnahme stellen reine Textelemente dar, d.h. Elemente, die nicht von Tags eingeschlossen sind, diese werden ohne Umweg dargestellt.

7 Realisierung

Der Abschnitt Realisierung beschäftigt sich mit der Umsetzung der Anwendung. Er umfasst Bildschirmfotos der Anwendung, eine Beschreibung der zusätzlich zu den nativen J2ME Bibliotheken eingesetzten Ressourcen sowie Besonderheiten die bei Test und Implementierung aufgetreten sind.

7.1 Eingesetzte Ressourcen

7.1.1 J4ME

J4ME ist eine Komponentensammlung für J2ME MIDP 2.0. Sie bietet Funktionalität für die Erstellung von User Interfaces, die Nutzung von GPS, extern wie intern, Logging sowie nativ nicht vorhandene Mathefunktionen wie Arcustangens.

Einsatz

Bei der Realisierung der Client-Anwendung wurde ein großer Teil des Userinterface auf Basis dieser Bibliothek erstellt. Durch die Quelloffenheit konnten Bedienelemente wie beispielsweise die Navigationsschaltfläche durch Copy und Paste und einigen Anpassungen schnell realisiert werden. Zudem erleichtert die Bibliothek die Darstellung von Informationen dadurch dass bereits viele der benötigten Steuerelemente, Textbereiche, Bilder und Formular-Felder und Ihre Steuerungslogik (Fokussierung, Eingabe) vorimplementiert sind.

7.1.2 KXML

KXML ist eine unter J2ME lauffähige Implementierung der sogenannten XML Pull API. Für den Umgang mit XML-Informationen gibt es verschiedene Ansätze: Pull und Push. Der Unterschied besteht in der Art wie die XML-Daten verarbeitet werden. Bei Push-Parsern wird vorher eine Verarbeitungsinfrastruktur aufgebaut, die anhand von Nachrichten über die verschiedenen XML-Elemente unterrichtet wird. Im Gegensatz dazu ist es bei Pull-Parsern nötig, explizit das nächste Element bzw. die nächste Struktur anzufordern. Das bedeutet die Abarbeitung des XML-Dokuments erfolgt analog zu seiner Struktur.

7.2 Bildschirmfotos

In diesem Abschnitt werden, zwecks verdeutlichung, Bildschirmfotos charakteristischer Situationen der Client-Software dargestellt.

Eingangsbildschirm Der Bildschirm, der zuerst auf dem Mobiltelefonbildschirm angezeigt wird. Auf ihm können z.B. Versionsinformationen angezeigt werden.

Hauptmenü Das Hauptmenü der Anwendung. Aktuell sind nur der Server Manager und das Optionsmenü enthalten.

Optionen In den Optionen wird das Verhalten der Anwendung bei bestimmten Ereignissen festgelegt. Aktuell lassen sich die Timeoutzeit, also wie lange bleibt ein Server in der Auswahl, die Transferpufferdimension und das Verhalten bei Erkennung eines neuen Servers einstellen.

Serverbenachrichtigung Eine Option die festlegt, wie auf die Erkennung eines Server reagiert werden soll. Mögliche Einstellungen sind Vibration, als Meldung oder beides.

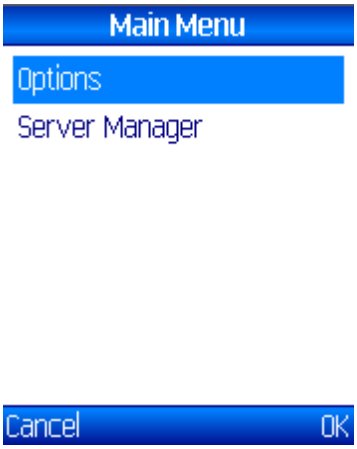
Gerenderte Seite Die Darstellung eines übertragenen Dokuments. Im oberen Bereich ist ein Logo zu sehen, im unteren Bereich befinden sich verschiedene Navigationsschaltflächen.

Plugin Auswahl Plugins entscheiden selbst, wie sie im gerenderten Dokument dargestellt werden. Die Standarddarstellung ist als auswählbare Schaltfläche.

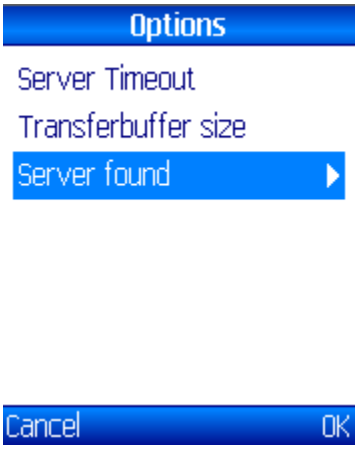
Direkte Verbindung Plugin Das Plugin für direkte Interaktion mit dem Server stellt in diesem Fall nur einen leeren Bildschirm dar. Es werden somit nur die betätigten Tasten an den Server übertragen und keine Rückmeldung empfangen.

Sicherheitsmeldung Bei nicht signierten MIDlets sind sensitive Operationen mit einer Sicherheitsabfrage verbunden, um den Anwender vor möglichem Schaden zu bewahren. Der Wortlaut ist hier abgebildet.

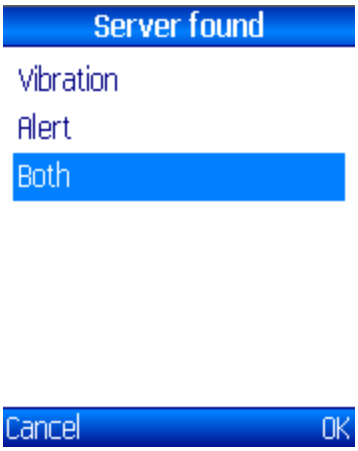
1. Eingangsbildschirm



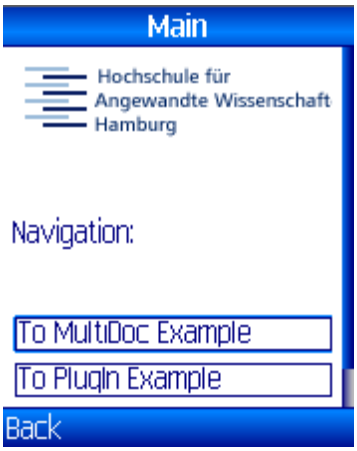
2. Hauptmenü



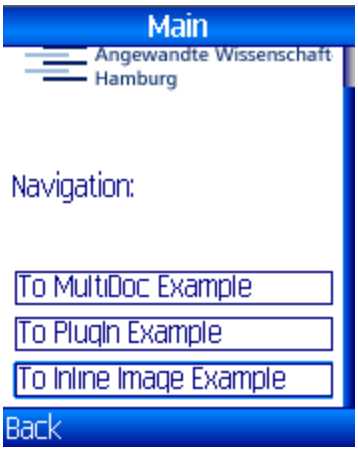
3. Optionen



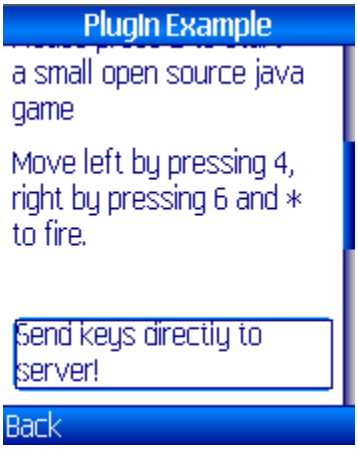
4. Serverbenachrichtigung



5a. Gerenderte Seite (oben)

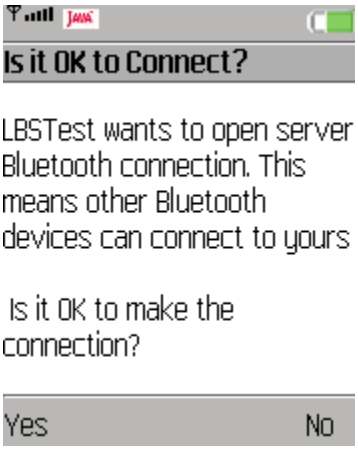


5b. Gerenderte Seite (unten)



6. Plugin-Auswahl

7. Plugin für direkte Serverbindung



8 Sicherheitsabfrage bei nicht signiertem Midlet

Tabelle 7.1: Client Bildschirmfotos

8 Schluss

8.1 Zusammenfassung

Im Rahmen dieser Arbeit, wurde eine verteilte Anwendung entwickelt um ortsbasierte Dienstleistungen stationär anzubieten oder mit Mobiltelefonen zu Nutzen. Als Proof of Concept wurden ein unter J2ME lauffähiger Client und für J2SE entwickelter Server entwickelt. Der Server erkennt die Anwesenheit des Clients anhand eines mit Bluetooth umgesetzten Annäherungsmessungs-Verfahren. Aufgrund der relativ geringen Reichweite von Bluetooth muss, da es ebenfalls für die Kommunikation eingesetzt wird, für jeden Point-Of-Interest ein Server bereitgestellt werden. Die für diesen Zweck konzipierte Architektur ermöglicht es ebenfalls, mit bestimmten Anpassungen ein unterschiedlich Strukturiertes Netz von Positionierung, Kommunikation und Informationsbereitstellung zu Nutzen. Als Beispiel sei hier GPS für die Positionierung und UMTS für die Kommunikation genannt

8.1.1 Funktionsumfang Client

Der Client hat die Möglichkeit, mit dem sich in Reichweite befindenden Server seiner Wahl Kontakt aufzunehmen, und Informationen zu beziehen beziehungsweise zu übermitteln. Diese mit XML strukturierten Informationen werden über ein HTTP ähnliches ebenfalls XML basiertes Protokoll ausgetauscht. Die Beispielimplementierung bietet auf dem Client Funktionalität für die Darstellung von folgenden Elementen: Dokumente und Unterdokumente (in einer Datei), Bilder (eingebettet und auch extern), Text und Navigationschaltflächen

8.1.2 Funktionsumfang Server

Die Serverkomponente stellt dem Client im Dateisystem vorliegende Inhalte zur Verfügung. Die Art der Client-Anfrage bestimmt dabei, wie diese verarbeitet wird. Es wurden zwei verschiedene Verbindungstypen implementiert. Zum einen das bekannte Anfrage-Antwort-Verfahren, bei dem vom Client eine Verbindung aufgebaut und eine Anfrage gestellt wird, und

der Server nach Beantwortung dieser die Verbindung wieder beendet. Zum zweiten eine direkt Verbindung, bei der die Verbindung, nach Initialisierung durch den Client, erst durch ein bestimmtes, vom Client empfangenes, Steuersignal durch den Server unterbrochen wird.

Je nach Verbindungstyp können unterschiedliche Anfrageverarbeiter zum Einsatz kommen. Zu Test- und Demonstrationszwecken der direkten Verbindung wurde eine Kopplung mit einem Open-Source-Java-Spiel geschaffen um die Nutzung eines Mobiltelefons als direktem Eingabegerät zu verdeutlichen.

Auf Grund von Zeitmangel, wurde auf die Implementierung von einem verteilten Anforderungsverarbeitungs-System verzichtet, ebenso wie auf die Server zu Server Interaktion. Dies macht den Einsatz von sogenannten Globalen-Informations-Servern, die dem Client Informationen im Push-Prinzip bei jedem Kontakt übermitteln unmöglich.

8.1.3 Funktion

Während des Ablaufs der Software treten aktuell noch vereinzelt Probleme auf. So passiert es teilweise, dass der Server den Client erst im zweiten Anlauf lokalisieren kann. Da bisher nur mit einem Server und einem Client getestet wurde ist das Verhalten bei mehreren Geräten gänzlich unbekannt. Diese Probleme sollten aber durch weitere Tests und Prüfungen beseitigt werden können. Die Software ist also sowohl für Client als auch für Server in keiner Weise als Vollständig anzusehen.

8.1.4 Fazit

Aufgrund des hohen Zeitdrucks konnten nicht alle konzipierten Funktionen umgesetzt und die Software nicht ausreichend getestet werden. Der bisher entwickelte Prototyp hat in ersten Tests allerdings seinen Nutzen aufgezeigt. Retrospektiv muss besonders die Nutzung von Bluetooth kritisch betrachtet werden. In der aktuellen Version sind für eine Nutzung als Kommunikationsmedium bzw. Lokalisierungsdienstleister noch zu viele Schwierigkeiten vorhanden. Besonders die lange Laufzeit bei der Suche von Geräten und die „Erblindung“, also die nicht Auffindbarkeit durch andere Geräte, während dieser Suche stellen Einschränkungen dar, die in so einem Projekt nur schwer zu kompensieren sind. Diese Nachteile wiegen die Vorteile der hohen Verbreitung wieder auf. Es bleibt abzuwarten, inwiefern sich die oben genannten Einschränkungen ins Bessere kehren mit den anstehenden Änderungen im Bluetooth 3.0 Standard. Sollte keine Besserung eintreten ließe sich konzipierte Architektur mit wenig Aufwand auf z.B. die Nutzung von WLAN umbauen, das in immer mehr Mobiltelefonen zum Einsatz kommt. Abschließend lässt sich sagen, dass das vorgestellte Konzept der Nutzung der I/O-Fähigkeiten der Mobiltelefone mit ortsbasierten Dienste ein so hohes

Potential bietet, dass eine Weiterführung der Arbeit auf diesem Gebiet sehr anstrebenswert ist.

8.2 Perspektive

Im Folgenden werden kurz einige Mögliche Erweiterungen und Verbesserungen vorgestellt.

8.2.1 Darstellung

Für den Clientteil der Anwendung bieten sich eine Vielzahl möglicher Erweiterungen an. Die Neuentwicklung der Darstellungsroutinen ermöglicht eine gute Eingriffsmöglichkeit. Es wäre denkbar statt der althergebrachten Methode der Darstellung von Informationen in die dritte Dimension zu wechseln. Moderne Mobiltelefone bringen bereits die nötigen 3D-Fähigkeiten mit. Dadurch ließen sich die Daten auf eine neue möglicherweise bessere Weise anordnen.

Denkbar wäre auch der Umstieg von dem selbst entwickelten Dokumententyp auf das XML-Mobile-Profile. Dies würde eine Steigerung der Kompatibilität zu normalen Mobiltelefon-Webportalen mit sich bringen.

8.2.2 Sicherheit

Besonders wichtig für die Akzeptanz der Software und den Einsatz beim Austausch sensibler Daten ist die Sicherheit derer. Wie bereits beschrieben wurde bei der Anwendung zwecks Vereinfachung auf den Einsatz von der Bluetooth eigenen Verschlüsselung verzichtet. Um diesen Mangel zu beheben ließen sich auf Protokollebene die Daten absichern. Bibliotheken, die die Nötige Funktionalität bieten, gibt es bereits, siehe (Bouncy Castle Team, 2008).

8.2.3 Test

Aufgrund fehlender Hardware, wurden die Tests der Client-Software zum Großteil auf einem Typ Mobiltelefon durchgeführt. Um allerdings die korrekte Funktionalität sicherzustellen wäre es nötig, diese Tests auf Geräten aller großen Hersteller zu wiederholen und die Resultate zu verifizieren. Weiterhin sollte das Verhalten der Software bei einer hohen Anzahl von Geräten getestet werden.

8.2.4 Peer-2-Peer

Eine interessante Möglichkeit, um ein umfassenderes Netz von Verfügbaren Zugangspunkten zu erhalten, wäre der Einsatz von Peer-2-Peer. Das bedeutet, dass das Clientgerät die Möglichkeit bekommt, als mobiler Server (ähnlich dem Server von UbiZoo) zu fungieren. Je nach Ausstattung und Konnektivität könnte so eine Art geroutetes Netzwerk entstehen das die Teilnehmer mit einer Vielzahl weiterer Informationen versorgt.

8.2.5 Internationalisierung

Aufgrund von Zeitmangel wurden die Menüs und Nachrichten in Englisch verfasst. Für einen besseren Zugang sollte eine Lokalisierung stattfinden.

8.2.6 Positionierung

Um einen höheren Detailgrad bei der Positionierung zu erhalten wäre es denkbar, eine Hardware zu entwickeln, die via Bluetooth mit Mobiltelefonen verbunden werden kann, um vorhandene Positionierungsinfrastrukturen nutzen zu können. Zum Beispiel (Gregor, 2006)

8.2.7 PlugIns für Endgeräte

Um mit dem sich ständig erweiternden Funktionsumfang von Mobiltelefonen Schritt zu halten, sollten neue PlugIns für die neuen Funktionen entwickelt werden. Dies hilft dabei Nutzer weiter für die Anwendung zu begeistern.

Literaturverzeichnis

[Apple 2007] APPLE: Apple - Nike + IPod. 2007. – URL http://www.apple.com/ipod/nike/gear.html

[Apple Inc 2008] APPLE INC: Apple I-Phone. 2008. – URL http://www.apple.com/iphone/

[Association 2003] ASSOCIATION, GSM: SE.23: Location Based Service. GSM Association, 2003

[Bass u. a. 2003] BASS, Len ; CLEMENS, Paul ; KAZMAN, Rick: Software Architecture in Practive - Second Edition. Addision Wesley, 2003. – ISBN 0-321-15495-9

[BlueCove Team 2008] BLUECOVE TEAM: BlueCove - JSR 82 Project. 2008. – URL http://www.bluecove.org/

[Bouncy Castle Team 2008] BOUNCY CASTLE TEAM: Bouncy Castle. 2008. – URL http://www.BouncyCastle.org/

[Bundesamt für Sicherheit in der Informationstechnik 2003] BUNDESAMT FÜR SICHERHEIT IN DER INFORMATIONSTECHNIK: Gefährdung von Drahtloser Datenübertragung. Bundesamt für Sicherheit in der Informationstechnik, 2003

[Ericsson 2008] ERICSSON: Ericsson. 2008. – URL http://www.ericsson.com/

[Google 2008] GOOGLE: Google Maps. 2008. – URL http://maps.google.com/

[Gregor 2006] GREGOR, Sebastian: Entwicklung einer Hardwareplattform für die Ermittlung von Positionsdaten innerhalb von Gebäuden. HAW-Hamburg, 2006

[Group 2007] GROUP, Bluetooth Special I.: Industry Statistics. Bluetooth SIG, 2007

[GSM Association 2008] GSM ASSOCIATION: Global System for Mobile Communication. 2008. – URL http://www.gsmworld.com/index.shtml

[Handurukande u. a. 2005] HANDURUKANDE, Sidath B. ; GANGULY, Samrat ; BHATNAGAR, Sudeept: Fast Bluetooth Service Discovery for Mobile Peer-to-Peer Applications. NEC Laboratories America, Inc., 2005

[HAW Hamburg 2008] HAW HAMBURG: Fachbereich Informatik der Hochschule für angewandte Wissenschaften in Hamburg. 2008. – URL http://www.informatik.haw-hamburg.de

[Humboldt-Universität zu Berlin 2008] HUMBOLDT-UNIVERSITÄT ZU BERLIN : MagicMap. 2008. – URL http://www2.informatik.hu-berlin.de/rok/MagicMap/index.htm

[JCP 2000] JCP: Mobile Information Device Profile (JSR-37) JCP Specification 1.0a. Sun Microsystems, Inc, 2000

[JCP 2001] JCP: PIM Optional Package 1.0 Specification. Sun Microsystems, Inc, 2001

[JCP 2003] JCP: Connected Limited Device Configuration 1.1 Specification. Sun Microsystems, Inc, 2003

[Kamida 2008a] KAMIDA: Kamida. 2008. – URL http://www.kamida.com/

[Kamida 2008b] KAMIDA: Socialight. 2008. – URL http://socialight.com/

[KAYWA 2008] KAYWA: KAYWA Reader. 2008. – URL http://reader.kaywa.com/

[Keles u. a. 2008] KELES, Fatih ; FALKENBERG, Eike ; NAPITUPULU, Jan ; SCHMIDT, Thomas: UbiZoo. HAW-Hamburg, 2008. – URL http://www.ubizoo.de/blog/

[Küpper 2005] KÜPPER, Axel: LOCATION-BASED SERVICES - Fundamentals and Operation. Wiley, 2005. – ISBN 0-470-09231-9

[Mark Weiser 2008] MARK WEISER: Homepage Mark Weiser. 2008. – URL http://www.ubiq.com/weiser/

[Michael Juntao Yuan 2003] MICHAEL JUNTAO YUAN: Let the mobile games begin. 2003. – URL http://www.javaworld.com/javaworld/jw-02-2003/jw-0221-wireless.html

[Microsoft 2007] MICROSOFT: Microsoft Imagine Cup. 2007. – URL http://imaginecup.com/

[Microsoft 2008a] MICROSOFT: DOT.NET Compact Framework. 2008. – URL http://msdn.microsoft.com/en-us/netframework/aa497273.aspx

[Microsoft 2008b] MICROSOFT: Windows Compact Edition. 2008. – URL http://www.microsoft.com/windows/embedded/products/windowsce/default.mspx

[Muller 2001] MULLER, Nathan J.: Bluetooth. mitp, 2001. – ISBN 3-8266-0738-4

[Open Mobile Alliance 2008] OPEN MOBILE ALLIANCE: Open Mobile Alliance. 2008. – URL http://www.openmobilealliance.org/

[Palm Inc. 2008] PALM INC.: Palm Operating System. 2008. – URL http://www.palm.com/

[Rene Melzer 2008] RENE MELZER: Area Mobile News: Bluetooth S.I.G: WLAN als Datenbeschleuniger für Nahfunk. 2008. – URL http://www.areamobile.de/news/8662.html

[Schmidt u. a. 2003] SCHMIDT, Albrecht ; BEIGL, Michael ; GELLERSEN, Hans-W.: There is more to context than Location. Telecooperation Office (TecO), University of Karlsruhe, 2003

[SKYHOOK Wireless, Inc 2008] SKYHOOK WIRELESS, INC: SKYHOOK Wireless. 2008. – URL http://www.skyhookwireless.com/

[Statistisches Bundesamt 2007a] STATISTISCHES BUNDESAMT: Statistisches Jahrbuch 2007. Statistisches Bundesamt Deutschland, 2007

[Statistisches Bundesamt 2007b] STATISTISCHES BUNDESAMT: Zahl der Woche - 80-Prozent-Marke bei der Handy-Ausstattung überschritten. Zahl der Woche. 2007. – URL http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Presse/pm/zdw/2007/PD07__019__p002.psml

[Sun Microsystems 2000] SUN MICROSYSTEMS: JSR-30: J2ME Connected, Limited Device Configuration. 2000. – URL http://jcp.org/aboutJava/communityprocess/final/jsr030/

[Sun Microsystems 2003] SUN MICROSYSTEMS: JSR 197: Generic Connection Framework Optional Package for the J2SETM Platform. 2003. – URL http://jcp.org/aboutJava/communityprocess/final/jsr197/

[Sun Microsystems 2004] SUN MICROSYSTEMS: JSR 75: PDA Optional Packages for the J2METM Platform. 2004. – URL http://jcp.org/aboutJava/communityprocess/final/jsr075/

[Sun Microsystems 2006] SUN MICROSYSTEMS: JSR 82: JavaTM APIs for Bluetooth. 2006. – URL http://jcp.org/aboutJava/communityprocess/final/jsr082/

[Sun Microsystems 2007] SUN MICROSYSTEMS: JSR-000139 Connected Limited Device Configuration 1.1. 2007. – URL http://jcp.org/aboutJava/communityprocess/final/jsr139/

[Sun Microsystems 2008a] SUN MICROSYSTEMS: Java Community Process. 2008. – URL http://java.sun.com/javame/technology/jcp.jsp

[Sun Microsystems 2008b] SUN MICROSYSTEMS: Java ME Connected Device Configuration FAQ - What is the difference between CDC and CLDC? 2008. – URL http://java.sun.com/javame/technology/cdc/faqs.jsp#cp7

[Sun Microsystems 2008c] SUN MICROSYSTEMS: Java ME Device Table. 2008. – URL http://java.sun.com/javame/technology/index.jsp

[Sun Microsystems 2008d] SUN MICROSYSTEMS: Java ME Technology. 2008. – URL http://java.sun.com/javame/technology/index.jsp

[Sun Microsystems 2008e] SUN MICROSYSTEMS: Java Plattform Micro Edition. 2008. – URL http://java.sun.com/javame/index.jsp

[Symbian Limited 2008] SYMBIAN LIMITED: Symbian Operating System. 2008. – URL http://www.symbian.com/

[v. Tanenbaum 2007] TANENBAUM, Marteen v.: Distributed Systems Principles and Paradigms. Pearson Education, Inc., 2007

[Tometa Software 2006] TOMETA SOFTWARE: J2Me vs. .Net. 2006. – URL http://www.tometasoftware.com/J2Me_vs_Net_whitepaper.asp

[UbiComp 2008] UBICOMP: Tenth International Conference on Ubiquitous Computing. 2008. – URL http://www.ubicomp.org/ubicomp2008/

[Ubisense Ltd 2008] UBISENSE LTD: UbiSense. 2008. – URL http://www.ubisense.de/

[Welte und Meriac 2006] WELTE, Harald ; MERIAC, Milosch: Project Sputnik. 2006. – URL http://events.ccc.de/congress/2006/Fahrplan/events/1736.en.html

[Wireless Application Protocol Forum, Ltd. 2001] WIRELESS APPLICATION PROTOCOL FORUM, LTD.: XHTML Mobile Profile. 2001. – URL http://www.wapforum.org/

Glossar

GCF Generic Connection Framework - Ein in der CLDC definiertes Framework zu vereinfachung des Zugriffs auf Verbindungen.

GIS Geographisches Informationssystem.

GPS Global Positioning System - ein geostationäres Satelliten Netz anhand dessen durch Time difference of arrival-Analyse Positionen auf der Erdoberfläche berechnet werden können

J2EE Java 2 Enterprise Edition

J2SE Java 2 Standard Edition

JCP Java Community Process - Prozess der standardisierung von Funktionalität für die Java Umgebung.

KVM Kilo Virtual Machine - Erste Referenz Implementierung einer J2ME-Virtual Machine. Ersetzt durch die HotSpot-Implementierung

P/Invoke => Eine in DOT.NET Framework definierte Methode zum Zugriff auf Funktionen außerhalb der Virtual Machine.

POI Point of interest - Sinngemäß übersetzt: Ort von Interesse. Einsatz zum Beispiel in Navigationssystemen: Tankstellen oder Rastplätze.

Transceiver Eine technische Gerätschaft die sowohl Daten empfangen (Receive) und übermitteln (Transmit) kann. In diesem Zusammenhang äquivalent mit drahtlos Transceiver.

XHTML EXtended Hypertext Markup Language. Eine Anwendung von XML mit der die Struktur von HTML-Dokumenten nachgebildet wird.

A Beispieldokumente

A.1 DTD

A.1.1 Informationsformat

Listing A.1: Inhalts Typ Definition

```
<!-- DTD für Informationsaustausch-Format -->

<!-- Gültige Elemente -->
<!-- *** Document *** -->
<!ELEMENT document (document| image| inputbutton| plugin| #PCDATA)*
    >
<!ATTLIST document
    title                                           CDATA #REQUIRED
>

<!-- *** image *** -->
<!ELEMENT image (#PCDATA)>
<!ATTLIST image
        text                                        CDATA #IMPLIED
        source                                      CDATA #IMPLIED
>

<!ELEMENT inputbutton (#PCDATA)>
<!ATTLIST inputbutton
        type                                        CDATA #REQUIRED
        value                                       CDATA #REQUIRED
>

<!ELEMENT pluginParameter (#PCDATA) >
<!ATTLIST pluginParameter
        name                                        CDATA #REQUIRED
```

```
26          value                                   CDATA #REQUIRED
27  >
28  <!ELEMENT plugin (pluginParameter)*>
29  <!ATTLIST plugin
30          name                                    CDATA #REQUIRED
31
32
33  >
```

A.1.2 Protokoll

Listing A.2: Protokoll Typ Definition

```
1  <!-- DTD für Protokoll -->
2
3  <!ELEMENT message (head,body)>
4  <!ELEMENT head (bodysize,connectiontype,messagetype,target?,source
       ?)>
5
6  <!ELEMENT bodysize #CDATA>
7  <!ELEMENT connectiontype #CDATA>
8  <!ELEMENT messagetype #CDATA>
9  <!ELEMENT target #CDATA>
10 <!ELEMENT source #CDATA>
11
12 <!ELEMENT body ANY>
```

A.2 Indexdokument

Listing A.3: Einstiegsseitencode

```
1  <document title="Main">
2          <image text="" source="hawlogo.png"></image>
3          Navigation:
4          <inputbutton type="navigate" value="example_subdoc.xml">To
               MultiDoc Example</inputbutton>
5          <inputbutton type="navigate" value="example_plugin.xml">To
               PlugIn Example</inputbutton>
```

```
6         <inputbutton type="navigate" value="example_inlinebinary.
              xml">To Inline Image Example</inputbutton>
7 </document>
```

A.3 Beispiele

A.3.1 Binärdaten

Listing A.4: Beispiel für integrierten Binärcode

```
1 <document title="Inline␣Image">
2 The following image is contained in the file. It's␣binary␣data␣is␣
      BASE64␣encoded␣and
3 stored␣directly␣between␣the␣image␣tags.
4
5 <image␣text="">iVBORw0KGgoAAAANSUhEUgAAADAAAAAwCAYAAA...
      AAAAASUVORK5CYII=</image>
6
7 <inputbutton␣type="navigate"␣value="index.xml">To␣Main</inputbutton
      >
8 </document>
```

A.3.2 Plugin

Listing A.5: Beispiel für die Nutzung von Plugins

```
1 <document title="PlugIn␣Example">
2         This page demonstrates the use of a plug-in to directly
              communicate
3         with the server.
4         Manual:
5         Please press 2 to start a small open source java game
6         Move left by pressing 4, right by pressing 6 and * to fire.
7         <plugin name="sendkey"></plugin>
8
9         Another Plugin Example:
10        This plugin allows you to start the integrated Camera take
              a
```

```
11          picture and save it on the server in the upload folder.
12          <plugin name="jsr135" width="480" height="600" />
13
14          <inputbutton type="navigate" value="index.xml">To Main</
                inputbutton>
15
16  </document>
```

A.3.3 Dokumentenstruktur

Listing A.6: Beispiel für mehrere Dokumente in einer Datei

```
1   <document title="Subdocument␣Example">
2           This page demonstrates the use of multiple documents in one
                file.
3           <inputbutton type="navigate" value="#0">UbiComp</
                inputbutton>
4           <inputbutton type="navigate" value="example_plugin.xml">To
                Plugin Example</inputbutton>
5           <inputbutton type="navigate" value="index.xml">To Main</
                inputbutton>
6           <document title="UbiComp">
7                   Ubiquitous Computing
8                   <inputbutton type="navigate" value="#-1#-1">To
                        Subdocument Example</inputbutton>
9                   <inputbutton type="navigate" value="#1#-1">To
                        RoboSoccer</inputbutton>
10          </document>
11          <document title="RoboSoccer">
12                  Roboter Soccer
13                  <inputbutton type="navigate" value="#-1#-1">To
                        Subdocument Example</inputbutton>
14                  <inputbutton type="navigate" value="#0#-1">To
                        UbiComp</inputbutton>
15          </document>
16  </document>
```

B Weitere Konfigurationen

Übersicht über mögliche Komponenten. Für jede Komponente ist eine eigene Implementierung nötig. Zudem ist wichtig, in wie fern mobile Endgeräte diese Komponenten unterstützen.

B.1 Mögliche Positionsquellen (Position originator)

- Global
  - GPS
  - Galileo
- Lokal
  - Proprietäre Systeme (z.B. IMAPS (Gregor, 2006) oder UbiSense (Ubisense Ltd, 2008))
  - WLAN-Basierte Systeme (z.B. MagicMap (Humboldt-Universität zu Berlin , 2008))
  - RFID basierte Systeme (z.B. OpenBeacon (Sputnik) (Welte und Meriac, 2006))

B.2 Drahtlos Kommunikationsmethoden

- UMTS, GPRS, HSDPA, GSM
- Bluetooth
- WLAN
- UWB (Ultra Wide Band)
- ZigBee

Versicherung über 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, 17. August 2008	
Ort, Datum	Unterschrift