TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät für Informatik
Institut für Systemarchitektur
Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill

DIPLOMARBEIT

Qualitätsbewusste Bereitstellung von Standortinformationen durch flexible und intelligente Kombination verschiedener Ortungstechnologien

Vorgelegt von: Christian Epperlein
geboren am: 5. Mai 1982 in Schlema

zum Erlangen des akademischen Grades
Diplom Medieninformatiker
(Dipl.-Medieninf.)

Betreueung:
Dr.-Ing. Waltenegus Dargie (Intern)
Forschungsfeld Mobile and Ubiquitous Computing

Dipl.-Kaufm. Lars Vogel (Extern)
T-Systems Multimedia Solutions GmbH

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Systemarchitektur, Professur Rechnernetze

AUFGABENSTELLUNG FÜR DIE DIPLOMARBEIT

Name, Vorname: Epperlein, Christian
Studiengang: Medieninformatik 2002
Matr. Nr.: 2928162

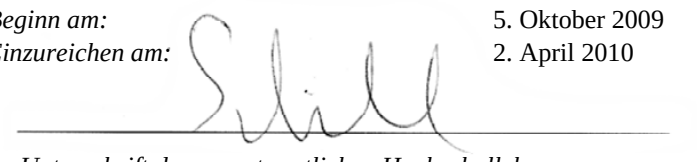
Thema: „Qualitätsbewusste Bereitstellung von Standortinformationen durch flexible und intelligente Kombination verschiedener Ortungstechnologien."

Steigende Verbreitung zunehmend leistungsfähigerer mobiler Endgeräte mit verschiedensten technologischen Möglichkeiten zur Standortbestimmung erhöhen die Möglichkeiten und das Potential für die Entwicklung standortbezogener Anwendungen. Eine solche Anwendung bietet entsprechend der aktuellen Position des Anwenders kontextbezogene Funktionalitäten. Zum Beispiel kann ein Messenavigator interessante Stände auffinden und den Weg dahin führen oder eine Touristenführung liefert Informationen über in der Nähe befindliche Sehenswürdigkeiten. Moderne Endgeräte verfügen über vielfältige Möglichkeiten, den Positionskontext zu ermitteln, z. B über die Handy-Funkzelle (GSM), GPS, WLAN, RFID bis hin zu Beschleunigungs- oder Lichtsensoren. Jedes dieser Verfahren hat verschiedene Eigenschaften bezüglich z. B. Ortungsgenauigkeit, Ortungsgeschwindigkeit und Verfügbarkeit. GPS zum Beispiel erlaubt nach längerer Initialisierungsphase die bis auf wenige Meter genaue kontinuierliche Standortbestimmung, zuverlässig jedoch nur außerhalb von Gebäuden. Durch Kombination verschiedener Ortungstechnologien und Verfahren lässt sich die Qualität und Zuverlässigkeit der Positionsdaten erheblich verbessern. Für eine LBS-Anwendung spielt es prinzipiell keine Rolle, welche Technologien zur Standortbestimmung verwendet wurden, sondern nur ob und mit welcher Qualität Standortinformationen genutzt werden können. Die Vielfältigkeit von Endgeräteeigenschaften, Positionierungstechnologien und Nutzeranforderungen erschweren die Anwendungsentwicklung maßgeblich. Um dem entgegenzutreten soll eine flexible Middlewarekomponente entwickelt werden, welche zur Verfügung stehende Positionierungstechnologien überwacht, auswertet und anbindet, und die ermittelten Standortdaten zusammen mit Aussagen über Qualität und Zuverlässigkeit über eine Schnittstelle bereitstellt. LBS-Anwendungen können damit Standortinformationen abfragen und dynamisch ihren Funktionsumfang anpassen und müssen sich nicht mit den endgerätespezifischen Details der Ortsbestimmung auseinandersetzen.

Forschungsschwerpunkte:

- Analyse und Vergleich der Stärken und Schwächen aktueller Ortungstechnologien und möglicher Anwendungsszenarien.
- Untersuchung von State-of-the-Art im Bereich Location-based Services, Location-Awareness und Location-middleware in Hinblick auf dynamische Aggregation von Ortungstechnologien/Sensoren
- Entwurf einer flexiblen Middleware-Komponente zur intelligenten Aggregation verschiedener Ortungstechnologien und passender Schnittstelle zur Bereitstellung der Positionsdaten und Qualitätsinformationen für LBS-Anwendungen.
- Prototypische Umsetzung mit ausgewählten Ortungstechnologien/Sensoren und Demonstration an Beispielanwendung.
- Evaluation der Ergebnisse.

Betreuer:	Dr.-Ing. Waltenegus Dargie (Intern)
	Dipl.-Kaufm. Lars Vogel (Extern)
Verantwortlicher Hochschullehrer:	Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill
Institut:	Institut für Systemarchitektur
Beginn am:	5. Oktober 2009
Einzureichen am:	2. April 2010



Unterschrift des verantwortlichen Hochschullehrers

Verteiler: 1 x Prüfungsamt, 1 x HSL, 1 x Betreuer, 1 x Student

TECHNISCHE UNIVERSITÄT DRESDEN

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage dem Prüfungsausschuss der Fakultät Informatik eingereichte Diplomarbeit zum Thema

Qualitätsbewusste Bereitstellung von Standortinformationen durch flexible und intelligente Kombination verschiedener Ortungstechnologien

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Dresden, den 30.06.2010

Unterschrift

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation und Problemstellung

Drahtlose Kommunikation wird immer mehr zum wesentlichen Bestandteil unseres täglichen Lebens. Satellitenkommunikation, Mobilfunk, WLANs, Wireless Sensor Networks (WSN) sind nur einige der drahtlosen Technologien, die heute jeden Tag genutzt werden. Sie erleichtern uns das Leben, indem sie uns überall und zu jeder Zeit in Verbindung halten [SGG08]. Mit modernen Smarthpones, den mobilen Alleskönnern, entstanden aus der Symbiose von klassischen Mobiltelefonen und PDAs[1], sind die Voraussetzungen für eine Vielzahl von mobilen Anwendungen geschaffen. Dazu zählen auch die standortbezogenen Dienste, englisch: location-based Services, kurz LBS. Ihre Popularität nimmt zu, wie man an aktuellen Statistiken erkennen kann. So interessiert sich zum Beispiel laut [Hus09a] ein Drittel der mobilen Nutzer für Navigationsdienste und mobile LBS. Sechs Prozent der europäischen mobilen Nutzer haben bereits GPS-/Navigationsdienste auf ihren Mobiltelefonen genutzt. Bei Nutzern von Smartphones mit uneingeschränktem mobilen Internetzugriff ist die Nutzung noch höher.

LBS zählen zu den kontextbezogenen Diensten. Das heißt, die bereitgestellten Informationen und Daten werden ausgewählt, zusammengestellt und aufbereitet unter Beachtung des individuellen Nutzerkontexts. Diese Kontextinformationen lassen sich aus der aktuellen Situation, den Aktivitäten, der Umgebung, der genutzten Technologie und den Vorgaben des Nutzers ableiten. Neben Zeitpunkt (z. B.: Tageszeit, Wochentag, jahreszeitlicher Monat) oder Profileinstellungen (z. B.: im Meeting, im Auto, zu Hause) spielt für LBS vor allem der Standort des Nutzers die primäre Rolle. In Abbildung 1.1 sind verschiedene Kontextparameter dargestellt, wobei zusätzlich in primären und sekundären Kontext unterschieden wird.

Grundlage für LBS ist der Standort bzw. Position, also die räumliche Lage eines mobilen Nutzers wie sich beispielsweise aus der Definition von [BL09] ablesen lässt:

> *Location-based services (LBS) are the delivery of data and information services where the content of those services is tailored to the current or some projected location and context of a mobile user. [BL09]*

[1] Personal Digital Assistent (Persönlicher digitaler Assistent)

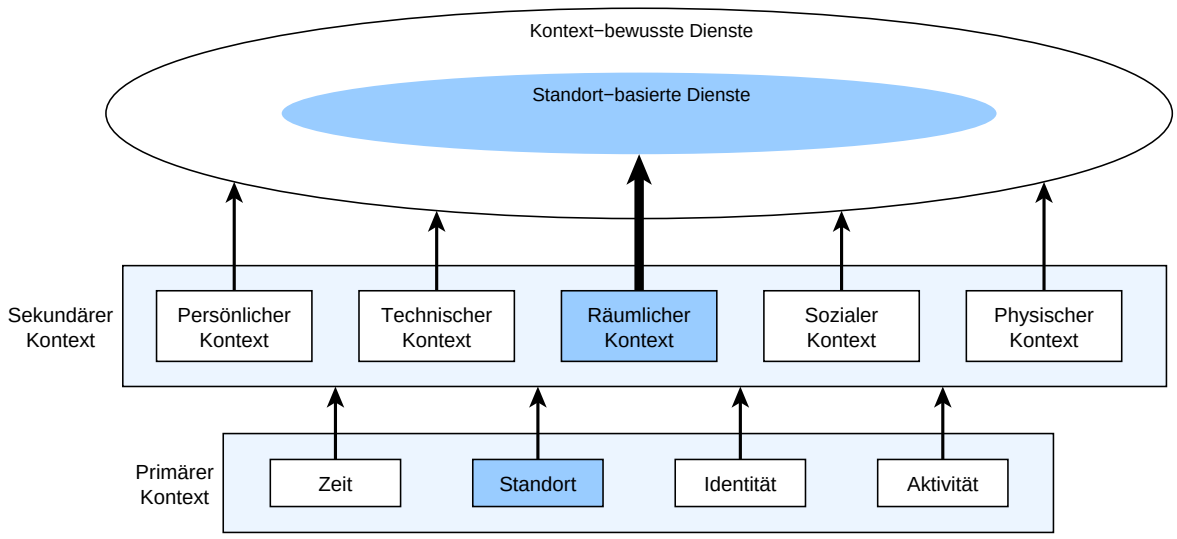


Abb. 1.1: Kontext-bewusste und Standort-bezogene Dienste [Küp05]

Beispiel

Ein Beispiel für eine typische und populäre LBS-Anwendung ist ein Fremdenführer (engl.: tourist guide). Ein Tourist ist in einer unbekannten Stadt unterwegs und beabsichtigt eine Mahlzeit einzunehmen. Mit Hilfe einer auf seinem mobilen Endgerät lauffähigen LBS-Anwendung könnte er eine Anfrage absenden, um zu erfahren, wo sich das nächste Restaurant befindet. Dazu wird zunächst automatisch die Position des Nutzers ermittelt, um die in der Nähe befindlichen Restaurants zu bestimmen. Das Ergebnis hängt also mit dem aktuellen Standort zusammen. Des Weiteren sollen nur Restaurants ausgegeben werden, die zum Zeitpunkt der Anfrage auch tatsächlich geöffnet haben. Deshalb wird der implizite Kontext des Zeitpunktes hinzugezogen, in dem die aktuelle Uhrzeit mit den Öffnungszeiten der gefundenen Restaurants abgeglichen wird. Als explizite Kontextinformationen könnte der Nutzer beispielsweise den Typ des Restaurants spezifizieren (z. B.: chinesische, französische oder gut bürgerliche Küche), wodurch das Ergebnis nochmals eingeschränkt wird.

1.1.1 Standortbestimmung

Das besondere an LBS ist, dass der Standort des mobilen Nutzers, oder vielmehr der Standort des von ihm genutzten Endgerätes, automatisch bestimmt und nicht manuell vom Nutzer eingegeben wird, beispielsweise durch Markierung eines Punktes auf einer Landkarte oder Eingabe einer

Adresse. Für die Bestimmung des Standortes (Ortung oder Lokalisierung) stehen eine Vielzahl an alternativen oder ergänzenden Technologien und Lösungen zur Verfügung, wie zum Beispiel GPS, Mobilfunk, WLAN oder Bluetooth.

Jede Ortungstechnologie hat verschiedene Eigenschaften. Die Voraussetzungen zum Einsatz richten sich nach der technologischen Verfügbarkeit, sowohl auf dem Endgerät des Nutzers (z. B. GPS-Empfänger, WLAN-Schnittstelle, Mobilfunk-Schnittstelle) als auch der Infrastruktur in der Umgebung des Nutzers (WLAN-Accesspoints, Sicht zu GPS-Satelliten, Mobilfunkbasisstationen). Als wichtigste Qualitätseigenschaft wird die Genauigkeit angesehen, mit welcher der Standort bestimmt werden kann, das heißt wie viele Meter genau die ermittelte Position mit tatsächlichen Position erwartungsgemäß übereinstimmt. Die Zeitspanne von der Ortungsanfrage bis zur Verfügbarkeit der ermittelten Position (Latenz), Energieverbrauch und Rechenleistung stellen weitere wichtige Parameter da. Die Sicherung des Datenschutzes der ermittelten Standortinformationen spielt ebenfalls eine große Rolle.

Vor allem die Einsatzfähigkeit einer Technologie innerhalb eines bestimmten Gebietes und die zu erreichende Qualität sind wichtige Kriterien für die Auswahl. Je nach Umgebung variieren die Eigenschaften und erreichbare Qualität. GPS zum Beispiel, lässt sich zuverlässig nur außerhalb von Gebäuden einsetzen. Ortung basierend auf WLAN hingegen lässt sich am besten innerhalb von Gebäuden oder in dichten Stadtbereichen einsetzen.

1.1.2 Problem

Die Komplexität und die Unterschiede in Eigenschaften wie Qualität und Einsatzfähigkeit der Ortungstechnologien führen zu einer Reihe von Problemen. LBS wissen nicht welche konkrete Ortungsmethode ihre Anforderungen am besten erfüllt, das heißt, welche Methode das beste Verhältnis aus Genauigkeit, Latenz, Energieverbrauch und Funktionalität bietet. Darüber hinaus wissen LBS nicht, welche Methoden am Einsatzort überhaupt verfügbar sind, oder vom Endgerät des Nutzers unterstützt werden. Die den Anforderungen entsprechende, passende Lösung zur Bestimmung des Standortes ist aber eine notwendige Voraussetzung, um das Potential von LBS-Anwendungen auszuschöpfen. Erschwerend kommt hinzu, dass der mobile Nutzer bei Ausführung von LBS oftmals in Bewegung ist. Die Umgebung kann sich also ändern, was sich auf die nutzbaren Ortungstechnologien auswirkt und einen Wechsel bedingen kann. Auch hier stehen LBS vor dem Problem der Erkennung und Auswahl der richtigen Methode.

LBS-Anforderungen

Um das Potential von LBS ausschöpfen zu können, muss eine Vielzahl an Anforderungen erfüllt werden:

Umfangreicher Nutzerkreis Um mit LBS einen großen Kreis an Nutzern erreichen zu können, welche mit technologisch sehr unterschiedlichen Endgeräten ausgerüstet sind, müssen verschiedene und alternative Ortungstechnologien unterstützt werden.

Vereinfachte Anwendungsentwicklung Das Auswahlproblem erschwert die Softwareentwicklung und -Wiederverwendung von LBS. Dabei ist die Auseinandersetzung mit den Details heterogener Ortungstechnologien, die Auswahl, oder das Zuschneiden auf konkrete Technologien entsprechend Szenarien und Anwendungen überhaupt nicht die Aufgabe von LBS.

Effektive Ressourcennutzung Mobile Endgeräte verfügen nur über eine begrenzte Akkuleistung und Rechenleistung. Ein ressourcenschonender Umgang, nicht nur der Anwendungen, sondern auch der genutzten Ortungstechnologien ist daher besonders wichtig. Durch dynamische Auswahl der optimalen Ortungsmethode lässt sich das beste Verhältnis zwischen Anforderungen der LBS-Anwendung (Latenz und Genauigkeit) und Aufwand der Standortbestimmung (Energieverbrauch, Rechenleistung, Netzwerkbelastung) erreichen.

Werden mehrere Standortbezogene Dienste gleichzeitig ausgeführt könnten im ungünstigsten Fall verschiedene Ortungstechnologien gleichzeitig aktiviert werden, was einen unnötigen Ressourcenverbrauch darstellt. Einmal ermittelte Standortinformationen sollten unter verschiedenen LBS geteilt werden. Auswahl der Ortungsmethode könnte sich nach der Anwendung mit den höchsten Anforderungen richten, welche für die anderen Anwendungen ebenso ausreichend sind.

Komplexe LBS. Beschränkung auf eine Ortungstechnologie kann implizit auch die Komplexität und den Einsatzbereich einer LBS-Anwendung einschränken, zum Beispiel den fließenden Übergang von draußen nach drinnen (outdoor nach indoor). Keine Ortungstechnologie liefert optimale Ergebnisse, unabhängig davon, in welcher Umgebung sich ein Nutzer befindet. Die wechselnden Eigenschaften und Anforderungen bedingen einen nahtlosen Wechsel der Ortungstechnologie, zum Beispiel von GPS auf WLAN, wenn ein Nutzer ein Gebäude betritt.

Quality of Service Wichtig für eine LBS-Anwendung ist, Informationen über die Qualität, vor allem Genauigkeit, der lieferbaren Standortinformationen zu erhalten. Das gilt vor allem dann, wenn die Anforderungen nicht erfüllt werden können. Die Anwendung kann dann auf die Abweichungen reagieren und zum Beispiel den Funktionsumfang eingrenzen, anstatt ungenaue oder falsche Standortdaten zu nutzen, was sich negativ auf die Nutzerakzeptanz auswirken kann. Qualitätsinformationen müssen gesammelt und in standardisierter Form zur Verfügung gestellt werden
Einhaltung des Datenschutzes ist ebenfalls äußerst wichtig. Mit den ermittelten Standortinformationen muss sorgsam umgegangen werden. Der Nutzer muss jederzeit wissen was mit seinen

Standortinformationen geschieht, und er muss beeinflussen können wann und an wen sie möglicherweise übertragen werden.
APIs zum Bezug von Standortinformationen, wie zum Beispiel das Mobile Location Protocoll (MLP)[2] oder JSR179[3], mangeln an Möglichkeiten, für Quality-of-Service-Parameter oder flexible Datenschutzmodelle [KJ06, S. 70].

Erweiterbarkeit Die Festlegung auf eine oder mehrere konkrete Ortungstechnologien behindert die Möglichkeiten zur Erweiterung. Zur Nutzung zukünftiger, bisher unbekannter Technologien müssten dann Änderungen an LBS-Anwendungen vorgenommen werden. Das "ultimative Ziel" [BCG08] ist die Realisierung einer einfachen LBS-Portabilität bezüglich verschiedener und erst zur Laufzeit dynamisch ausgelieferter Ortungslösungen.

Aggregation Die Qualität der Standortbestimmung lässt sich unter Umständen steigern, wenn mehrere Verfahren miteinander kombiniert werden. Die ermittelten Standortdaten verschiedener Ortungstechnologien könnten verschmolzen werden, was durch die enthaltene Redundanz zu einer höheren Genauigkeit und Zuverlässigkeit führt. Eine andere Möglichkeit ist die sogenannte Koppelnavigation, bei der GPS und mechanische Sensoren (sogenannte MEMS[4]) kombiniert werden, so dass auch bei kurzzeitigem Sichtverlust zu den Satelliten eine Navigation fortgeführt werden kann. Mechanische Sensoren (Lage, Beschleunigung, Kompass) sind heute bereits in vielen Smartphones verfügbar[iSu09]. Es sollte möglich sein, nicht nur einzelne Ortungstechnologien, sondern auch verschiedene Kombinationsverfahren integrieren, auswählen und nutzen zu können.

1.1.3 Lösungsansätze und Abgrenzung

Die Erfüllung der vielen Anforderungen, welche an die Bestimmung des Standortes gestellt werden um das Potential von LBS auszuschöpfen stellt eine große Herausforderung dar. Folgende Lösungsansätze wurden bisher verfolgt:

Manuelle Auswahl Ein Ansatz, der eher einer Umgehung des Problems entspricht, besteht darin, in Frage kommende Ortungslösungen im Voraus auszuwählen und die LBS-Anwendung und ggf. die Infrastruktur dafür anzupassen. So wird zum Beispiel für Outdooranwendungen typischerweise GPS eingesetzt und Indooranwendungen mit einer WLAN-Lösung verbunden. Dieser klassische Ansatz, welcher in Abbildung 1.2a skizziert wird, vereinfacht zwar die LBS-Entwicklung stellt allerdings einen Kompromiss dar, welcher mit massiven Einschränkungen vieler der oben genannten Anforderungen einher geht.

[2] http://www.openmobilealliance.org/Technical/release_program/mlp_v31.aspx (Stand: 29.06.2010)
[3] http://jcp.org/en/jsr/detail?id=179 (Stand: 29.06.2010)
[4] Micro-Electro-Mechanical Systems

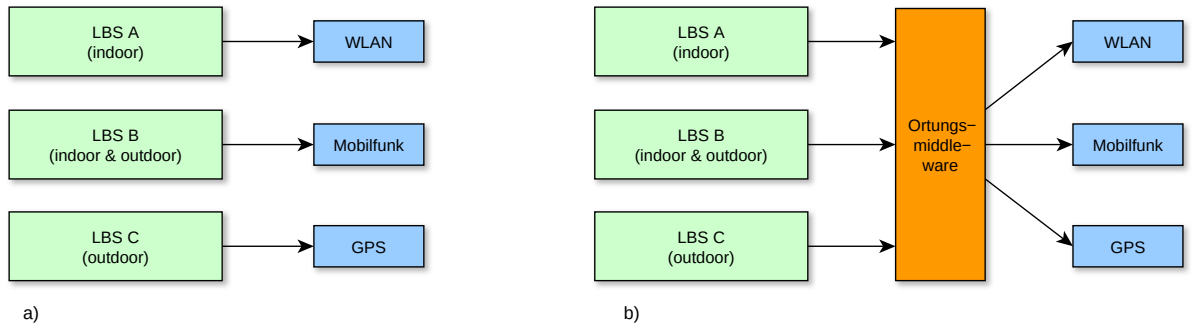


Abb. 1.2: Klassischer Ansatz (a) und Middleware-Ansatz (b)

Hybride Ortungsverfahren Um die Schwächen einzelner Ortungsverfahren zu umgehen, können hybride Lösungen entwickelt werden, welche verschiedene Ortungsverfahren integrieren. Ein Beispiel ist Assisted-GPS (A-GPS), bei welchem die Standortbestimmung durch zusätzliche Assistenzdaten beschleunigt und zuverlässiger gestaltet wird. Eine Möglichkeit dabei ist, Mobilfunkortung zur schnellen Bestimmung eine ersten, aber ungenauen Position zu verwenden (Vergleich Kapitel 3.3.2). Kommerzielle Anbieter wie Skyhook Wireless[5] oder Spotigo[6] haben Lösungen entwickelt, welche sowohl GPS, Mobilfunk als auch WLAN integrieren. Solche proprietären Lösungen stellen allerdings auch nur wieder einen Teilausschnitt der theoretischen Möglichkeiten zur Standortbestimmung dar und sind in Flexibilität und Erweiterbarkeit eingeschränkt.

Ortungsmiddleware Da, wie bereits angedeutet wurde, die Auseinandersetzung mit den Eigenschaften heterogener Ortungslösungen nicht Aufgabe der LBS-Anwendung ist, sollte Auswahl und Steuerung, von einer eigenen Instanz erfolgen, wie in Abbildung 1.2b) skizziert wird. Eine solche Softwarezwischenschicht wird auch als Middleware bezeichnet (siehe Kapitel 3.4). Middlewareansätze zur Standortbestimmung sind ein aktuelles Thema der letzten Jahre und haben zahlreiche Forschungsarbeiten sowie Standardisierungsversuche hervorgebracht. Diese Arbeit wird sich näher mit diesem Thema auseinandersetzen.

1.2 Ziele der Arbeit und Gliederung

Da die Standortbestimmung das Herz einer jeden LBS-Anwendung darstellt, ergibt sich LBS-Qualität primär auch aus Performanz und Eigenschaft der Standortbestimmung. [FBDH08]

[5] http://www.skyhookwireless.com (Stand: 29.06.2010)
[6] http://www.spotigo.com/ (Stand: 20.10.2009

Zusammenfassend würde eine Ortungsmiddleware die Flexibilität der Standortbestimmung und Ausführung von LBS deutlich verbessern, den Nutzerkreis erweitern, ressourcenschonenderen Umgang fördern und durch verbesserte Qualität die Zufriedenheit der Nutzer steigern.

Das Ziel dieser Arbeit liegt in der Analyse des aktuellen Forschungsstandes und Herausarbeitung und Entwicklung von Middlewareansätzen zur dynamischen Auswahl und Aggregation von heterogenen Ortungstechnologien und Ortungslösungen.

Dazu soll in Kapitel 2 zunächst näher auf LBS und die relevanten Komponenten eingegangen werden. Es müssen die Eigenschaften, Typen und Anforderungen von LBS-Anwendungen und der benötigten Standortinformationen analysiert werden.

Ein wichtiger Schritt ist die Analyse der Charakteristiken von Ortungstechnologien und Lösungen in Kapitel 3. Kenntnis der Wirkungsweise verschiedener Ortungslösungen ist eine wichtige Voraussetzung für die Untersuchung und Konzeptionierung von Ortungsmiddlewares.

Das Hauptziel der Arbeit besteht in der Analyse und Konzeption eines Ortungsmiddlewareansatzes. Kapitel 4 widmet sich daher der Anforderungsanalyse. Es werden einige Beispielszenarien vorgestellt und die notwendigigen Schritte abgeleitet, die von einer Ortundsmiddleware erfüllt werden müssen. Es sollen wichtige Aspekte und Schlüsselfragen untersucht werden, die bei der Konzeptionierung beachtet werden müssen. Schließlich werden die Anforderungen abgeleitet, welche an eine Ortungsmiddleware zu stellen sind. Das Kapitel schließt mit der Untersuchung des aktuellen Standes im Forschungs- Standardisierungs- und kommerziellen Bereich.

In Kapitel 5 wird schließlich das Konzept für einen Middlewareansatz ausgearbeitet und vorgestellt. Es soll eine Schnittstelle entwickelt werden, über welche LBS Standortdaten anfragen können. Die Ortungsmiddleware soll dann aus verschiedenen zur Verfügung stehenden Ortungsdiensten die optimale Lösung auswählen und aktivieren. Dazu müssen die Eigenschaften der Ortungsdienste betrachtet und mit den Anforderungen der LBS in Beziehung gesetzt werden. In Abbildung 1.3 wird diese Grundidee skizziert. Die LBS-Anwendung wird dadurch von der konkret zu benutzenden Ortungstechnologie getrennt und muss sich nur um Anfragen und Anforderungsspezifikation kümmern.
Ein besonderes Ziel dabei ist es, eine flexible und erweiterbare Architektur zu entwickeln, die sowohl um beliebige weitere Ortungslösungen erweiterbar ist, als auch um neue, verbesserte Algorithmen zur Auswahl der Ortungsdienste und Aggregation der gelieferten Standortinformationen.

Ein weiteres wichtiges Ziel ist die Einbeziehung des Quality-of-Service der Standortbestimmung. Auswahl und Aktivierung der Ortungsdienste soll nach Qualitätsanforderungen der LBS geschehen. Die tatsächlich erreichte oder theoretisch aktuell zu erreichende Ortungsqualität sollen ebenfalls an LBS zurückgeliefert werden, damit diese die erhaltenen Standortdaten “qualitätsbewusst” verarbeiten können.

Eine prototypische Implementierung soll die Funktionalität des konzeptionierten Ansatzes demonstrieren. Ein wichtiges Ziel dabei, ist die Umsetzung für ein modernes Smartphone, denn nur

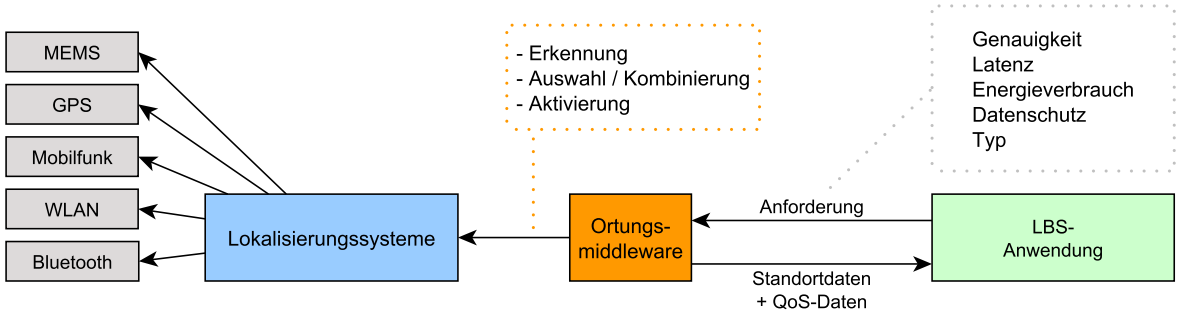


Abb. 1.3: Grundidee Ortungsmiddleware

so können praxisrelevante Erkenntnisse gesammelt werden. Kapitel 6 geht detailliert auf Spezifika der verwendeten Plattform und die Umsetzung des Konzept ein.

Zum Abschluss sollen in Kapitel 7 die Ergebnisse bewertet werden. Es soll gezeigt werden, welche Aspekte des Konzepts mit welchem Erfolg umgesetzt werden konnten. Außerdem sollen wichtige Erkenntnisse aus dem Verhalten der Middleware und der verwendeten Ortungsdienste vorgestellt werden.

Mit einer Zusammenfassung und einem Ausblick auf offene Aspekte und weiterführende Aufgaben und Möglichkeiten schließt die Arbeit.

2 Standortbezogene Dienste

2.1 Architektur

LBS basieren auf verschiedenen Technologien und umfassen viele Unterarchitekturen wie drahtlose Netzwerke, das mobile Internet und die verschiedenen Ortungstechnologien. Die folgenden Komponenten bilden das Grundgerüst von LBS (nach [BL09]):

Informationen über die reale Welt, in welcher sich der Nutzer befindet. Sie werden vom LBS-Anbieter auf den Standort des Nutzers zugeschnitten. Der Bezug erfolgt aus Datenbanken oder dem Internet. Die Informationsquellen werden durch verschiedene Inhaltsanbieter (Contentprovider) repräsentiert.

Endgerät des mobilen Nutzers wie z. B. Smartphone oder Netbook. Dient zur Ausführung von LBS-Anwendungen sowie als Grundlage der automatischen Standortbestimmung.

Lokalisierungssystem zur automatischen Bestimmung der geografischen Position des mobilen Endgeräts des Nutzers.

Drahtloses Netzwerk zur Kommunikation (Anfragen und Antworten) zwischen LBS-Nutzer und LBS-Anbieter, z. B. WLAN, GSM oder UMTS.

LBS-Anbieter und zugehörige Software, Dienste und Komponenten, die benötigt werden, um auf den Nutzer zugeschnittene Informationen aufzubereiten und zur Verfügung zu stellen.

2.2 Funktionseinheiten

Die Komplexität von LBS wird ersichtlich, wenn man einen genaueren Blick auf die LBS-Beschaffungskette wirft. Aus den verschiedenen Schritten von der Positionsbestimmung bis hin zur Generierung der LBS-Daten und den entsprechenden Funktionalitäten lassen sich typische Aufgaben (Roles) identifizieren. Zur Erfüllung dieser Aufgaben ist eine Vielzahl von Akteuren

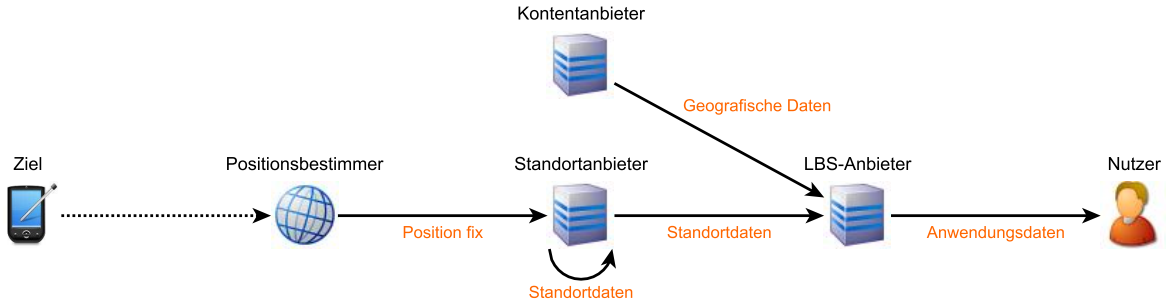


Abb. 2.1: Allgemeines Modell einer LBS-Beschaffungskette

(Actor) beteiligt, repräsentiert durch verschiedene Personen, Firmen oder Organisationen. Diese interagieren miteinander, bieten und konsumieren Dienste, und unterhalten die technische Infrastruktur von mobilen Endgeräten über Server bis hin zu komplexen Drahtlosnetzwerken.

Folgende Schritte sind in der LBS-Beschaffungskette auszuführen:

1. automatische[7] Lokalisierung
2. Verbesserung der reinen Positionsdaten zu gehaltvolleren Standortdaten
3. Assoziation der Standortdaten mit geografischem Inhalt oder anderen Standortdaten und Zuschneiden auf den Nutzer

Abbildung 2.1 zeigt ein allgemeines Modell einer LBS-Beschaffungskette. Die Interaktion zwischen zwei Aufgaben erfolgt über sogenannte Referenzpunkte und schließt Kommunikationsverbindungen, Schnittstellen, Protokolle, Verbindungsdienste ein.

Ziel (Target) ist das zu ortende, verfolgende, sichtende Objekt oder Individuum, ausgestattet mit mobilem Endgerät (Mobiltelefon, PDA, Smartphone, GPS-Empfänger, Badge) welches die Voraussetzungen zur Standortbestimmung erfüllt

Positionsbestimmer (Position originator) ermittelt die Position. Erfolgt bei gerätebasierten (und netzwerkunterstützten) Verfahren auf dem Endgerät des Ziels, bei netzwerkbasierten (und geräteunterstützten) Verfahren durch einen Verantwortlichen der Infrastruktur.

Standortanbieter (Location provider) ist eine zwischen Positionsbestimmer und LBS-Provider gelagerte Einheit. Sie veranlasst und kontrolliert die Positionsbestimmung im Auftrag des LBS-providers, sammelt Positionen für ein oder mehrere Ziele, erweitert die Positionsdaten zu gehaltvolleren "high-level"-Standortdaten und liefert sie an den LBS-Provider zurück. Dieser Dienst wird als Ortungsdienst (Location Service LCS) bezeichnet.

[7] automatisch heißt, durch ein technisches Verfahren und nicht durch manuelle Nutzereingabe

LBS-Anbieter ist die zentrale Rolle und implementiert die Geschäftslogik der LBS-Anwendung. Er sammelt Standortdaten von ein oder mehreren Zielen, analysiert und kombiniert sie mit anderen geografischen Daten und liefert die Ergebnisse an den LBS-Nutzer.

Content-provider unterstützt den LBS-Provider durch liefern von geografischen Daten wie Karten, Navigationsdaten oder Point of Interests (PoIs)

LBS Nutzer ist der Konsument der LBS-Anwendung und nutzt sie entweder mit seinem mobilem Endgerät oder von einem stationärem Endgerät wie dem PC.

Dieses Modell stellt nur einen allgemeinen Ansatz dar und demonstriert die zu Grunde liegenden Mechanismen von LBS. In der Realität kann es verschiedene Ausprägungen geben, je nachdem welche Aufgabe von welchem Akteur übernommen wird (Vergleich [Küp05, S.250ff]). Außerdem muss nicht jede Aufgabe von verschiedenen Akteuren übernommen werden. In *mobilen* LBS ist der LBS-Nutzer gleichzeitig das Ziel, d. h. er konsumiert LBS-Anwendungen auf Basis seines eigenen Standortes. Auch ist nicht immer ein Standortanbieter notwendig. Bei einer Navigationslösung mit GPS-Ortung werden die ermittelten Positionsdaten des GPS direkt auf dem mobilen Endgerät genutzt, während hingegen bei einer Ortung über Mobilfunk zum Beispiel auf eine Schnittstelle eines Drittanbieters zurückgegriffen werden kann, welcher wiederum verschiedene Mobilfunkbetreiber anspricht.

Eine wichtige Erkenntnis dieses Modells ist, dass es verschiedene Arten von Standortinformationen generiert und konsumiert werden. Es wird unterschieden in Position (position fix), Standort (location data), geografischem Inhalt (geographic content) und Anwendungsdaten (application data). Für diese Arbeit sind besonderers die Ausprägungen der Standortdaten von Bedeutung.

2.3 Eigenschaften und Anwendungsklassen

Aus den Unterschieden der verschiedenen LBS-Anwendungsklassen lassen sich typische Merkmale ableiten, welche für die für die Standortbestimmung relevant sind.

2.3.1 Pull oder Push

Die Übermittlung der Informationen eines LBS kann mit oder ohne Interaktion des Nutzers initiiert werden.

Bei *Pull-Diensten* (*reaktiv* bzw. *aktiv*) muss der Nutzer zunächst eine explizite Anfrage stellen, bevor Informationen übermittelt werden. Dies ähnelt dem Aufruf einer Website im Internet, bei dem zunächst die Adresse im Browser eingetragen wird. Pull-Dienste können weiter in *Funktionsdienste*, wie Absenden eines Notrufes, und *Informationsdienste*, wie Suche des nächstgelegenen Restaurants, unterschieden werden.

Push-Dienste (*proaktiv* bzw. *passiv*) übermitteln Informationen obwohl sie vom Nutzer nicht explizit oder nur indirekt (implizit) angefragt wurden. Die Übermittlung wird durch ein Ereignis gestartet, welches durch einen Zeitschalter oder das Betreten eines bestimmten Bereiches ausgelöst wird. Beispiel für einen indirekten Pull-Dienst könnte ein Benachrichtigungsdienst sein, bei dem sich ein Nutzer registrieren kann und anschließend passend zur aktuellen Stadt Informationen über stattfindende Veranstaltungen erhält. Werbenachrichten, die in bestimmten Bereichen eines Einkaufszentrums an den Nutzer übertragen werden sind ein Beispiel für nicht explizit angefragte Push-Dienste.

Da Push-Dienste nicht direkt an vorangegangene Nutzeraktionen gekoppelt sind, ist ihre Realisierung weitaus komplexer und schwieriger, als die von Pull-Diensten.

Für die Standortbestimmung ist folgendes von Bedeutung: bei Pull-Diensten wird die Lokalisierung wie die LBS-Ausführung erst mit Anfrage durch den Nutzer gestartet. Bei bereichssensitiven Push-Diensten hingegen muss bereits im Vorfeld eine kontinuierliche Lokalisierung erfolgen, um die Bereiche zu erkennen welche ein Ereignis auslösen sollen.

2.3.2 Selbstverfolgung oder Fremdverfolgung

Self-referencing oder *Cross-referencing*

Eine weiteres Unterscheidungsmerkmal ist, ob der Nutzer einer LBS-Anwendung sich selbst oder andere mobile Nutzer verfolgt, sei es, ebenfalls von einem mobilen Gerät oder einer festen Station.

Der erste Fall entspricht den typischen LBS-Anwendungen. Der Nutzer ist mobil unterwegs und konsumiert Informationen, bezugnehmend auf den ermittelten eigenen Standort, über sein mobiles Endgerät. Dazu zählen die Informationsdienste, Navigation, PoIs oder Gebührenerfassung.

Beim zweiten Fall werden andere mobile Nutzer verfolgt. Zum einen kann die Anwendung ebenfalls auf einem mobilen Endgerät abgerufen werden, zum Beispiel wenn man andere Nutzer finden will, die sich in der unmittelbaren Umgebung aufhalten und gleiche Interessen verfolgen. Zum anderen kann die Anwendung von einer festen Station abgerufen werden. Das klassische Flottenmanagement oder die Überwachung von Kindern oder Objekten sind Beispiele dafür. Dabei sind Datenschutzrichtlinien besonders wichtig. Es muss sichergestellt sein, dass der zu überwachende Nutzer auch ausdrücklich zugestimmt hat, geortet werden zu dürfen.

2.3.3 Einmalige oder kontinuierliche Ortung

Bei einmaliger Ortung (single) wird eine Anfrage gesendet, einmalig die Position bestimmt und auf dieser Basis eine gefilterte Antwort geschickt. Beispiel dafür ist der Restaurantfinder. Einmalige Ortungen werden meistens für Pull-Services benutzt.

Im Gegensatz dazu wird bei kontinuierlicher Ortung (periodic) die Position fortwährend in bestimmten Intervallen bestimmt und übermittelt. Dies ist zum Beispiel bei Echtzeitanwendungen wie Navigation notwendig. Bei Push-Services, welche als auslösende Ereignisse das Betreten eines bestimmten Bereiches verwenden, ist meist eine kontinuierliche Ortung notwendig, welche bei Start der Anwendung beginnt und überwacht, wann welcher Bereich betreten wird.

2.3.4 Typische Anwendungsklassen

LBS-Anwendungen lassen sich in typische Klassen einordnen, welche sich in den genannten Eigenschaften unterscheiden. Informationsdienste zum Beispiel wie lokale Wettervorhersage oder ÖPNV-Informationen versorgen den Nutzer mit Informationen zu seinem aktuellen Standort. Sie haben Pull-Charakter, beziehen sich auf die eigene Position.

Typische LBS-Kategorien mit Beispielen sind in Tabelle A.1 aufgelistet (basierend auf [Rot05, SGG08]).

2.4 Standortinformationen

Grundlage für LBS sind Standortinformationen über einen oder mehrere Nutzer.

2.4.1 Eigenschaften

Die Position (*Position*) oder der Standort (*Location*) beziehen sich auf einen physischen Platz in der realen Welt. Es kann grundsätzlich unterschieden werden in geografischen und beschreibenden sowie absoluten und relativen Standort, wie in Tabelle 2.1 gegenübergestellt ist.

	absolut	relativ
geografisch	Koordinaten (z.B. Länge, Breite, Höhe)	Entfernung zu Koordinaten
beschreibend	Objektbezeichnung (z.B. Gebäude, Etage, Raum)	Entfernung / Lage zu Objekten

Tab. 2.1: Klassifikation Standort

Der geografische Standort entspricht einem einzelnen Punkt im Euklidischen Raum und wird durch zwei- oder dreidimensionale Koordinaten ausgedrückt. Diese Information allein ist allerdings

sehr benutzerunfreundlich, wenn der Bezug zu umliegenden Objekten hergestellt werden soll, zum Beispiel der Aufenthalt im Raum eines Gebäudes.

Der beschreibende (oder auch logische, symbolische) Standort bezieht sich auf ein geografisches Objekt wie Land, Stadt, Straße, Gebäude, Raum welches durch einen Namen, Bezeichner oder Nummer referenziert wird. Dies entspricht eher dem fundamentalen Konzept des alltäglichen Lebens.

Das Ergebnis der Lokalisierung kann sowohl in der einen wie auch der anderen Form vorliegen, weshalb Abbildungsmechanismen zwischen den beiden Formen notwendig sind.

Absolute Angaben beziehen sich auf ein Referenzsystem, zum Beispiel die Koordinaten im geodätischen Referenzsytems WGS-84[8] oder eine Postanschrift. Relative Angaben beziehen sich hingegen auf Lagebeziehung zu anderen Standorten. Also zum Beispiel Entfernungen zu anderen Koordinaten, oder Zimmer eines bestimmten Gebäudes.

Die Begriffe Position (*Position*) oder Standort (*Location*) werden häufig synonym verwendet, stellen aber bei genauerer Betrachtung verschiedene Daten dar. Die Position, auch *position fix* (feste Position) genannt, ist das direkte Ergebnis der Ortung. Sie entspricht den rohen Positionsdaten des Zielobjekts in einer vom angewandten Ortungsverfahren abhängigen Darstellungsform, meist in Form von reinen Koordinaten. Erst durch Verknüpfung der Positionsdaten mit weiteren standortbezogenen Informationen wie Geschwindigkeit, Richtung, Übertragung in ein anderes Referenzsystem und Hinzufügen von Daten wie Identifikation des Zielobjekts und Qualitätsinformationen, werden gehaltvollere Standortdaten erzeugt. Diese sind wesentlich geeigneter für LBS-Anwendungen, vor allem wenn Unabhängigkeit von der konkret genutzten Lokalisierungsmethode gefordert wird. [Küp05]

Die Standortinformationen können aus folgenden Daten bestehen:

- geografische Position (nicht zwingend in Originalform der Ortung)
- logische/symbolische Position (z. B.: Bereich, Gebäude, Stadt, Zimmer, Stockwerk)
- Zeitcharakter (aktuell, initial, zuletzt bekannt)
- Ausrichtung (der Bewegung)
- Geschwindigkeit (der Bewegung)
- Qualität (Genauigkeit, Zuverlässigkeit, Aktualität)
- Identität (des Ziels und Identitätstyp z. B. MSISDN, IMSI, IP-Adresse, Name, Pseudonym, MAC-Adresse)

[8] World Geodetic System 1984 [wgs00]

2.4.2 Qualität

Informationen darüber zu haben, welche *Qualität* die von einem Ortungsdienst gelieferten Standortinformationen haben und diese an einen LBS zu übermitteln, hat eine große Bedeutung. Nicht immer können die Anforderungen des LBS auch tatsächlich erfüllt werden. Er muss also über dieses Problem informiert werden, um entsprechend reagieren zu können, in dem zum Beispiel der Funktionsumfang eingrenzt oder der Anwender informiert wird. Weitergabe von Daten auf Basis ungenauer oder falscher Ortsinformationen kann zu verminderter Nutzerakzeptanz führen.

Wichtige Parameter für die Qualität von Standortinformationen sind Genauigkeit, Aktualität und Zuverlässigkeit. Sie werden maßgeblich von den Eigenschaften der verwendeten Bestimmungsmethoden beeinflusst, auf welche in Kapitel 3.2 detaillierter eingegangen wird.

2.4.2.1 Genauigkeit

Die Genauigkeit gibt an, welche Größe der Bereich oder die Region hat, in der sich das Ziel laut Lokalisierung befinden soll. Man spricht auch von Auflösung, oder vom Abstand der ermittelten zur tatsächlichen Position. Eine Genauigkeit von beispielsweise 30 Meter bedeutet, dass sich der Nutzer in einem Kreis mit dem Radius 30 Meter um die ermittelte Position befindet.

In diesem Zusammenhang wichtig, und oftmals verwechselt ist der Begriff *Präzision*. Sie bezieht sich auf die Wahrscheinlichkeit, dass eine gegebene Genauigkeit erreicht wird und sich das Ziel auch tatsächlich innerhalb des gedachten Kreises befindet.

Beide Parameter lassen sich im Voraus durch Messungen ermitteln und müssen immer zusammen angegeben werden. In der Praxis wird zur Vereinfachung und besseren Vergleichbarkeit allerdings oft auf einen typischen Präzisionswert festgelegt. So werden Genauigkeitsangaben für GPS meist auf eine Präzision von 95% bezogen. Das heißt, eine GPS-Genauigkeitsangabe von 20 Meter bedeutet, dass sich der Nutzer mit einer Wahrscheinlichkeit von 0,95 im Umkreis mit Radius 20 Meter um die ermittelte Position befindet.

Erfolgt eine dreidimensionale Standortbestimmung, muss außerdem zwischen horizontaler und vertikaler Genauigkeit unterschieden werden.

2.4.2.2 Aktualität

Die Aktualität gibt Auskunft über die zeitliche Aussagekraft einer Standortinformation. Je weniger Zeit seit der Lokalisierung vergangen ist umso aktueller ist die Standortinformation. Je größer diese Zeit ist, umso unwahrscheinlicher ist es, dass sich der Nutzer noch an der ermittelten Position aufhält.

2.4.2.3 Zuverlässigkeit

Die Zuverlässigkeit gibt allgemein Auskunft darüber mit welcher Wahrscheinlichkeit ein Nutzer sich tatsächlich in der ermittelten Position befindet und leitet sich unter anderem aus der Aktualität und der Genauigkeit ab.

Je ungenauer die Standortbestimmung erfolgte und je weiter die Aktualität sinkt, um so niedriger ist die Zuverlässigkeit.

2.5 Mobile Endgeräte

Die Möglichkeiten zur Nutzung von mobilen LBS und die Voraussetzungen für die dafür notwendige Standortbestimmung hängen wesentlich vom verwendeten mobilen Endgerät ab.
Die Auswahl an unterschiedlichen Modellen und Typen ist groß, und eine genaue Abgrenzung kaum möglich. Zu groß die Vielfalt an Größen, Formen, Funktionen und technischen Eigenschaften. Mit fortscheitender Entwicklung der Technologien im Bereich Kommunikation, Hardware und Software werden mehr und mehr Features in ein Modell integriert und die Grenzen der unterschiedlichen Typen verschwimmen.

2.5.1 Typen

Untersucht man die Ursprünge der verschiedenen Geräte lassen sich aber dennoch einige Grundtypen identifizieren. Dazu zählen das klassische Mobiltelefon, das reine mobile Navigationssystem (PND), der persönliche digitale Assistent (PDA) das Notebook und das im Fokus dieser Arbeit stehende populäre Smartphone. [BL09]

(Todo: Bilderübersicht)

Mobiltelefon Das Mobiltelefon ist eines der meist genutzten mobilen Geräte und wurde ursprünglich konzipiert um unterwegs, “mobil” telefonieren zu können. Mit der aktuellen dritten Generation werden hochwertige Sprechverbindungen und Hochgeschwindigkeitsdatendienste ermöglicht. Obwohl primär für Sprache und Kurzmitteilungen (SMS) genutzt, unterstützt es eine Vielzahl an Diensten wie mobiles Internet, Videotelefonie, Email, digitale Photos, Videos und Musik. Neben der Anbindung an die verschiedenen Mobilfunknetze (GSM, UMTS) gehört inzwischen oft auch Bluetooth, Infrarot und sogar WLAN zu den verfügbaren Kommunikationsschnittstellen. Im Vergleich zu Smartphones wird meist ein proprietäres Betriebssystem verwendet, welches bis auf Java-Anwendungen, sogenannte Midlets, kaum um Funktionalitäten erweitert werden kann und eine eher starre Menüführung bietet.

PDA Der Personal Digital Assistent (auch Handheld, Palmtop oder Pocket PC) ist ein kompakter handlicher Computer, der primär für das persönliche Informationsmanagement, also der Verwaltung von Kontakten, Terminen, Aufgaben und Notizen konzipiert wurde. Berührungsempfindliche Displays und Handschriftenerkennung ermöglichen eine leichte Bedienbarkeit. Fortgeschrittene Rechenleistung und Datenübertragungsmöglichkeiten wie Bluetooth, WLAN oder Infrarot haben den PDA multimedia- und internetfähig gemacht und erlauben die Datensynchronisation mit Emailservern und stationären Computern.

Mobiles Navigationssystem PND (Personal Navigation Device oder Portable Navigation Device) ist ein transportables, elektronisches stand-alone Gerät welches Lokalisierung und Navigation vereint. Zur Ortung wird typischerweise ein GPS-Empfänger benutzt. Kabelgebundene (z. B. USB) oder drahtlose (z. B. Bluetooth) Schnittstellen ermöglichen den Bezug von Kartenmaterial, Aktualisierungen oder Verkehrsinformationen. In handlicher Form, mit großem, meist berührungssensitivem Display wird es zur Navigation im Auto, oder als Fußgänger eingesetzt. Die Verbreitung von Navigationssoftware für die anderen Endgerätetypen stellt eine ernst zu nehmende Alternative dar. Laut einer Studie von iSuppli [iSu07] sollen PNDs bis 2014 von GPS-fähigen Smartphones verdrängt werden.

Notebook Das Notebook, bzw. das noch kleinere und mobilere Netbook, entspricht dem mobilen Ersatz für den Büro- oder Heim-PC. Es zeichnet sich durch hohe Rechenleistung, ein großes, zuklappbares Display, eingebaute Tastatur, Touchpad als Mausersatz, vielfältige Anschlussmöglichkeiten z. B. für USB-Geräte, externen Monitor, Audiogeräte und optisches Laufwerk aus. WLAN gehört inzwischen ebenfalls zum Standard. Kartenschnittstellen ermöglichen den Zugang zum Mobilfunknetz. Größe, Gewicht, vor allem aber die leistungsbedingte nur einige Stunden anhaltende Akkulaufzeit begrenzen die Mobilität.

Smartphone Aus der Konvergenz in Funktionsumfang und Einsatzgebiet von Mobiltelefon und PDA hat sich die Entwicklung des Smartphones herauskristallisiert. Es vereint die Vorteile von Beiden und stellt damit Funktionen wie Telefongespräche, Textnachrichten (SMS, Email), mobiles Internet, als auch persönliches Informationsmanagement (Kontakte, Termine, Aufgaben, Notizen) und Anwendungen auf einem Gerät zur Verfügung. Populäre Vertreter sind das iPhone von Apple, die N-Serie von Nokia, oder das Blackberry von RIM

Zu den typische Merkmalen zählen:

- in Funktionalität erweiterbares Betriebssystem, meist von einem Drittanbieter z. B. Android, Symbian OS, Windows Mobile, Linux, iPhone OS, Palm OS, BlackBerry OS
- einem PDA ähnelnder Formfaktor, meist in Hochkant und Querformat bedienbar.

- großes, hochauflösendes, oft berühungssensitives Display (Touchscreen)
- QWERTZ-Tastatur, ausklappbar oder digital über berührungssensitives Display
- Umfangreiche Schnittstellen zur Kommunikation und Datensynchronisation z. B. Mobilfunk (GSM/UMTS), WLAN, Bluetooth, Infrarot, USB
- Digitalkamera
- integrierter GPS-Empfänger

2.5.2 Technologietrends und Fokus

Mobilfunkortung, GPS, WLAN und Bluetooth

Im Fokus dieser Arbeit stehen vor allem Smartphones, welche die Eigenschaften von Mobiltelefonen und PDAs verschmelzen und die Voraussetzungen erfüllen für die mobile Ausfühung von LBS-Anwendungen und die dafür benötigte Lokalisierung. Betrachtet man die Technologietrends, lässt sich ableiten, welche Möglichkeiten für die Standortbestimmung und Auslieferung von LBS zur Verfügung stehen.

GPS findet auch auf Mobiltelefonen und Smartphones immer mehr Verbreitung. Von den 2006 ausgelieferten mobilen Endgeräten waren 11,1% mit GPS ausgestattet. In 2014 sollen es 29,6% sein [iSu07].

Ein Einblick in den aktuellen Technologietrend lässt sich zum Beispiel aus der Übersicht auf teltarif.de[9] geben. Dort wurden Ende 2009[10] 316 verschiedene Dualband UMTS Endgeräte gelistet. Davon verfügeb bereits 118 (37%) über ein integriertes GPS-Modul.
Die am meisten verbreitete drahtlose Schnittstelle, neben der Anbindung an das Mobilfunknetz ist Bluetooth mit 97% der oben genannten Geräte. Über WLAN verfügen immerhin 40% der Geräte.
Von den 316 gelisteten Geräten verfügen 29% sowohl über WLAN, Bluetooth als auch GPS-Empfänger. Damit stehen also auf fast einem Drittel der zu diesem Zeitpunkt verfügbaren Mobiltelefone bereits vier verschiedene Technologien zur Verfügung, welche zur Standortbestimmung genutzt werden können.
Infrarot scheint eher rückläufig zu sein. Verfügen zwar 17,7 Prozent der 316 gelisteten Geräte über diese Schnittstelle, so Schrumpft der Anteil in Kombination von WLAN auf 6,9% und mit Bluetooth auf 2,5%. Alle vier Features werden nur von 2,2% der Geräte unterstützt.
In Abbildung 2.2 sind die Kombinationsmöglichkeiten in einer Übersicht zusammengestellt.

Ebenfalls von Bedeutung für die Standortbestimmung können Sensoren wie Accelerometer (Beschleunigungssensor) oder Magnetometer (Kompass) sein. Beschleunigungssensoren werden zurzeit

[9] http://www.teltarif.de/h/suche.html (Stand: 12.11.2009)
[10] Stand 12.11.2009

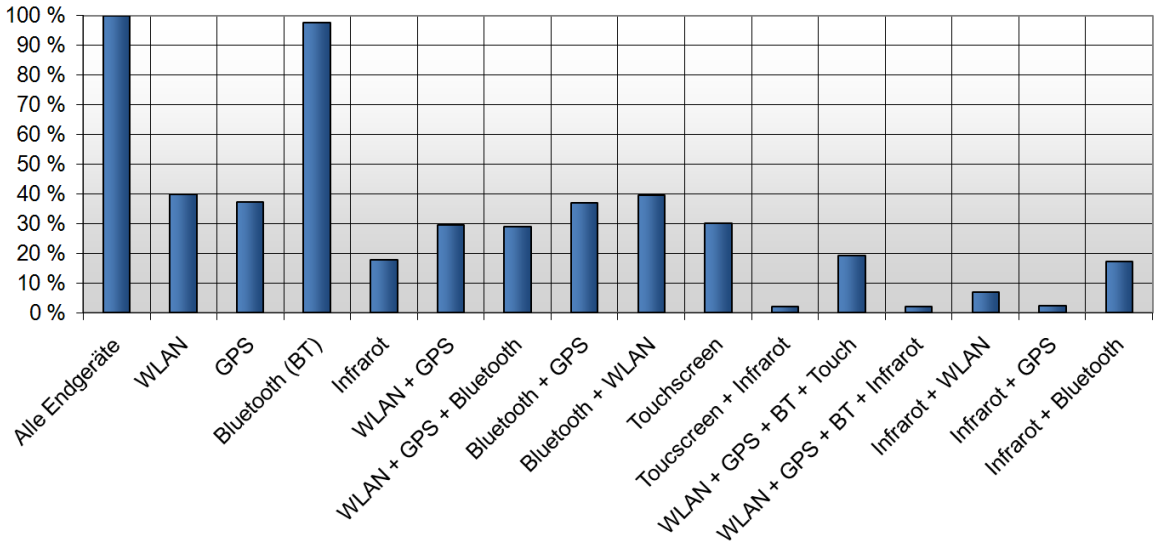


Abb. 2.2: Übersicht mobile Endgeräte und Ausstattung

vorrangig für Spiele oder die automatische Ausrichtung des Displays verwendet, spielen aber auch für das Energiemanagement eine Rolle. In 2010 sollen laut [iSu09] bereits ein Drittel der weltweit ausgelieferten Mobiltelefone mit Beschleunigungssensoren bestückt sein.

Für den Bezug der LBS-Inhalte als auch möglicher Kommunikation mit Standortbestimmungsservern wird eine Datenverbindung benötigt. Auch das mobile Internet wächst und erreicht den Massenmarkt. Transparente und erschwingliche Tarife tragen zur Nutzerakzeptanz bei. So soll zum Beispiel die Flatratenutzung in Westeuropa von 11% in 2009 auf 25% in 2014 steigen [Hus09b].

2.5.3 Grenzen

Rechen-, Speicher-, Energieressourcen

Obwohl Smartphones ein breites Spektrum an Funktionen und technologischen Voraussetzungen bieten, ist ihre Leistungsfähigkeit limitiert. Sie verfügen nur über begrenzte Rechen- und Speicherressourcen. Aufwendige Berechnungen, sowohl für die Ausführung von LBS als auch die notwendige Standortbestimmung sind daher nur in bestimmtem Maße möglich und werden daher oft auf Servern im Netzwerk durchgeführt, und die Ergebnisse später zum Endgerät zurückübertragen. Auch die niedrige Akkuleistung stellt eine bedeutsame Grenze dar.

Ein ressourcenschonender Umgang ist notwendig, um eine bestmögliche Nutzung des Endgerätes

und der Anwendungen zu gewährleisten. Dies stellt eine besondere Herausforderung an die Auswahl und Nutzung der Technologien zur Standortbestimmung dar.

2.6 Bezugsformen von LBS

Clientanwendung oder Webanwendung

Um eine LBS-Anwendung auf dem mobilen Endgerät verfügbar zu machen, gibt es zwei prinzipielle Möglichkeiten, sie auszuliefern. Entweder als Clientanwendung, welche vom Nutzer zunächst auf dem Endgerät installiert werden muss oder als Webanwendung, welche über den Browser des Endgerätes abgerufen wird. Beide Formen haben Vor- und Nachteile und verschiedene Einschränkungen was die Möglichkeiten der Standortbestimmung betrifft.

2.6.1 Clientanwendung

Eine Clientanwendung muss zunächst auf dem Endgerät installiert (und ggf. manuell aktualisiert) werden, was einen Aufwand für den Nutzer darstellt. Es müssen für die verschiedenen Betriebssysteme (z. B.: Symbian, Android, iPhone-OS) angepasste Anwendungen erstellt und ausgeliefert werden. Bei einigen Plattformen müssen Anwendungen über den jeweiligen Anbieter bezogen werden zum Beispiel iTunes Store [11] oder Android Market[12]. Eine Übernahme des Anbieters erfolgt allerdings oft erst nach Prüfung, was einen weiteren Nachteil darstellt.

Der Vorteil von Clientanwendungen liegt in der meist höheren Performance und besseren Gestaltungsmöglichkeiten und Integrierung in das Gerät. Außerdem besteht ein einfacher Zugriff auf viele Gerätefunktionen wie WLAN, GPS, Sensoren und damit auch die Möglichkeit vielfältige Ortungsverfahren auszunutzen.

2.6.2 Webanwendung

Bei einer Webanwendung erfolgt Bezug und Darstellung im Browser, unter Verwendung der verbreitetsten Webtechnologien wie HTML[13] + CSS[14] und Javascript[15]. Der Vorteil liegt darin, dass vom Nutzer keine Installation durchgeführt werden muss und immer die aktuellste Anwendung verfügbar ist. Außerdem gibt es weniger Einschränkungen durch Spezifika verschiedener Endgeräte, was die Wiederverwendbarkeit deutlich erhöht.

[11] http://www.apple.com/de/itunes/what-is/store.html (Stand: 29.06.2010)
[12] http://www.android.com/market/ (Stand: 29.06.2010)
[13] http://www.w3.org/TR/html401/ (Stand: 29.06.2010)
[14] http://www.w3.org/Style/CSS/ (Stand: 29.06.2010)
[15] http://www.w3schools.com/JS/ (Stand: 29.06.2010)

	Webanwendung	Clientanwendung
Clientbasierte Ortung (z. B. GPS, WLAN-Fingerprint)	nicht/kaum möglich	möglich
Netzwerkbasierte Ortung (z. B. Mobilfunk, WLAN-Cell-ID)	möglich	möglich
Performance / Usability	eher niedriger	eher höher
Installation	nicht notwendig (Darstellung im Browser)	notwendig (evtl Download über iTunes-Store/Androidmarket)
Updates	automatisch	manuell
Wiederverwendbarkeit	hoch	niedrig
Nutzung Gerätefunktionen	nicht/kaum möglich	möglich

Tab. 2.2: Vergleich Webanwendung und Clientanwendung

Ein großer Nachteil ist es allerdings, dass aus einer Webanwendung nicht, oder nur schwer, auf die Gerätefunktionen wie WLAN oder GPS zugegriffen werden kann. Deshalb gibt es eine deutliche Einschränkung der möglichen Ortungsverfahren. So lässt sich zum Beispiel der GPS-Adapter eines Endgerätes von einer Webanwendung kaum oder gar nicht nutzen. Deswegen wird vornehmlich auf netzwerkbasierte Verfahren wie Mobilfunkortung oder netzwerkseitige WLAN-Ortung zurückgegriffen.
Mit der GeoLocation-Api[16], Teil des kommenden HTML5[17], scheint sich eine Alternative abzuzeichnen. Dabei greift der Browser auf die Gerätefunktionen zu, führt die Ortung durch und stellt eine für Webanwendungen nutzbare standardisierte Schnittstelle zur Verfügung. Mehr dazu in Kapitel 4.4.2.2.

In Tabelle 2.2 sind die wichtigsten Vor- und Nachteile gegenübergestellt.

[16] http://dev.w3.org/geo/api/spec-source.html (Stand: 29.06.2010)
[17] http://dev.w3.org/html5/spec/spec.html (Stand: 29.06.2010)

3 Standortbestimmung

3.1 Grundlagen

Der Prozess zur Bestimmung der räumlichen Position eines Ziels wird als Standortbestimmung, Lokalisierung oder Ortung bezeichnet. Es gibt verschiedene Methoden und Verfahren die sich in ihren Eigenschaften teilweise unterscheiden. Grundlegend wird die Standortbestimmung durch die folgenden Elemente bestimmt (angelehnt an [Küp05]):

- ein oder mehrere durch Meßmethoden zu überwachende Parameter
- A-priori-Standortdaten von Referenzpunkten
- Methoden zur Positionsberechnung
- Beschreibendes oder räumliches Referenzsystem
- Infrastruktur
- Protokolle zur Koordination des Standortbestimmungsprozess

Zunächst müssen ein oder verschiedene, wahrnehmbare Parameter gemessen werden, wie z. B. Winkel, Reichweite, Entfernung, Geschwindigkeit. Durch diese Parameter wird die räumliche Lage des Ziels zu verschiedenen festen Punkten der Umgebung wiedergespiegelt, deren Koordinaten bereits bekannt sein müssen. Meist werden die physikalischen Eigenschaften von Funk-, Infrarot-, Ultraschallsignalen genutzt. Im Kontext der Ortung spricht man auch von *Pilotsignalen*.

Auf Basis der ermittelten Meßergebnisse und Koordinaten der festen Punkte wird anschließend die Position bestimmt. Die dafür zu benutzende Methode, z. B. Lateration oder Angulation, hängt vom Typ der überwachten Parameter ab. In Tabelle 3.1 sind die grundlegenden Methoden, die zu überwachenden Parameter und wie sie gemessen werden gegenübergestellt.

Eine Lokalisierungssmethode liefert die Position (position fix) des Ziels in Bezug auf ein beschreibendes (z. B.: ID der Mobilfunkzelle, Raumnummer, Etage) oder geografisches Referenzsystem (z. B Koordinaten in WGS-84-Format)

Neben dem Endgerät dessen Position bestimmt werden soll, spielen die Basisstationen eine wichtige Rolle, welche die Messungen selbst durchführen oder das Endgerät dabei unterstützen. Diese Basisstationen repräsentieren die oben erwähnten festen Punkte, deren Koordinaten bekannt

Methode	zu überwachende Parameter	gemessen durch
Nachbarschaftserkennung (Proximity sensing)	Cell-ID, Koordinaten	Wahrnehmung von Pilotsignalen
Lateration	Entfernung Entfernungsdifferenz	Signallaufzeit, Pfadverlust[18]der Pilotsignale, Signalstärke Differenz von Signallaufzeit, Pfadverlust
Angulation	Winkel	Antennenanordnung
Koppelnavigation (Dead reckoning)	Position und Richtung der Bewegung, Geschwindigkeit, Entfernung	Jede andere Methode Kreiselinstrument (Gyroskop) Beschleunigungssensor (Accelerometer) Wegmesser (Hodometer)
Mustervergleich (Pattern matching)	optisches Abbild Fingerprint	Kamera Signalstärke

Tab. 3.1: Grundlegende Methoden zur Positionsermittlung

sein müssen. Beispiele dafür sind Satelliten, Mobilfunkstationen, Wireless Access Points (WAP, deutsch “Funkzugangspunkt”).

3.1.1 Methoden

Nachbarschaftserkennung Auch *Proximity* oder *Cell of Origin* (CoO). Gelangt ein mobiles Gerät in Funkreichweite einer Station deren Standort bekannt ist, lässt sich aus dieser Nähe ein Rückschluss auf den Standort des Engderätes ziehen. Die Identifikation der Basisstation erfolgt über einen Identifikator, z.B. der Cell-ID einer Mobilfunkzelle oder der Macadresse eines WAP. Sind mehrere Stationen in Reichweite, lässt sich aus der Signalstärke ableiten, welche davon am nächsten ist.

Lateration Bei Lateration erfolgt die Positionsberechnung auf Basis der Entfernung zu bekannten Punkten. In der Ebene (Trilateration) sind dafür die Koordinaten von drei Punkten und der jeweilige Abstand zum Punkt, dessen Position berechnet werden soll, notwendig. Sind die Entfernungen des Endgerätes zu mindestens drei Basisstationen bekannt (abgeleitet z. B. aus Signallaufzeit oder Signalstärke), lässt sich daraus die Position des Endgerätes berechnen.

[18]Verlust der elektromagnetischen Leistung zwischen Sender und Empfänger

Angulation Bei Angulation erfolg die Berechnung der Position auf Basis der Winkel, statt Entfernungen.

Mustervergleich Auch *Patternmatching* oder *Fingerprinting*. Die Standortbestimmung erfolgt durch Abgleichen empfangener Signalcharakteristiken (z. B. Signalstärke der Funksignale) mit einer Datenbank. Diese sogenannten "*Fingerprints*" müssen vorher in der sogenannten "*Offlinephase*" eingemessen werden. Das heißt, es wird an verschiedenen Meßpunkten die Position und dort gemessenen Signale gespeichert. Bei WLAN-Ortung werden zum Beispiel die Signalstärken empfangener WAP zusammen mit der MAC-Adresse in der *"Radiomap*" genannten Datenbank abgelegt.

Koppelnavigation *Dead reckoning*. Bei der Koppelnavigation wird die fortwährende Ortung eines sich bewegenden Ziels durch Messung von Geschwindigkeit, Beschleunigung, Bewegungsrichtung unterstützt. So lässt sich zum Beispiel die Qualität der durch GPS ermittelten Positionsinformationen durch Richtungs- und Geschwindigkeitssensoren verbessern oder aufrechterhalten wenn die Sichtverbindung zu den Satelliten, z. B. in Tunneln, kurzzeitig unterbrochen ist.

3.1.2 Signalparameter

Die folgenden Parameter von Funksignalen werden für die Lokalisierung typischerweise analysiert:

Signallaufzeit *Time of Flight* (ToF), *Time of Arrival* (ToA), *Time Difference of Arrival* (TDoA). Elektromagnetische Wellen bewegen sich mit Lichtgeschwindigkeit. Mit dem Wissen der Geschwindigkeit und des Zeitunterschieds zwischen Senden und Empfangen des Signals, kann die Entfernung berechnet werden. Da Lichtgeschwindigkeit mit ca. 300000km/s sehr kurze Laufzeiten hat, sind exakte Zeitmesser notwendig.
Das gleiche Prinzip lässt sich auch auf langsamere Signale wie Ultraschall anwenden.

Signalrichtung *Angle of Arrival* (AoA) oder *Direction of Arrival* (DoA). Mit Hilfe von Richtantennen kann der Winkel des empfangenen Signals auf dem Endgerät bestimmt werden. Das ist auf Grund der Bewegung des Geräts allerdings nur ungenau. Eine andere Möglichkeit besteht darin, die Bauweise der Basistationen auszunutzen. Gibt es mehrere Antennen (gewöhnlich 2-4), die in verschiedene Richtungen strahlen, lässt sich der Umkreis der Funkzelle in Segmente aufteilen.

Signalstärke *Received Signal Strength* (RSS). Da die Signalstärke mit steigender Entfernung abnimmt lässt sich auch daraus der Abstand zu einer Basisstation berechnen. Im Vergleich zu

Laufzeitmessungen weisen diese Verfahren allerdings eine sehr viel höhere Fehlerrate auf, vor allem bei Satelliten und mobilfunkbasierten Methoden. Nur im Innenraumbereich, wo extrem kurze Entfernungen die Zeitmessung sehr schwierig gestalten, wird mitunter auf signalstärkebasierte Methoden zurückgegriffen.

3.2 Eigenschaften

Jede Lokalisierungsmethode und die dabei involvierten Technologien weißen eine vielzahl an Eigenschaften auf, welche die erreichbare Qualität der zu ermittelnden Standortinformationen nachhaltig beeinflussen. Eine hohe Qualität zeichnet sich, wie in Kapitel 2.4.2 angesprochen, durch hohe Genauigkeit, Zuverlässigkeit und Aktualität der ermittelten Position aus.

3.2.1 Bestimmungsort

Gerätebasiert, netzwerkbasiert, hybrid

Eine wichtige Eigenschaft einer Lokalisierungsmethode bezieht sich auf den Ort, an dem die eigentliche Standortbestimmung durchgeführt, und wie dies unterstützt wird.

Zunächst wird grundsätzlich in gerätebasierte und netzwerkbasierte Verfahren unterschieden [SSE06].

Bei gerätebasierten Verfahren erfolgt die Standortbestimmung lokal auf dem Endgerät. Signalmessung und Berechnungen zur Bestimmung der Position werden vom Empfänger durchgeführt, der sich auf dem mobilen Gerät befindet. Dazu ist keine Netzwerkverbindung notwendig, was zu wesentlich mehr Privatsphäre und Datenschutz führt. Allerdings ist der Energieverbrauch höher, vor allem wenn Signale schlechter empfangen werden (z. B. in Tunneln oder Innenraum) Die bekanntesten Vertreter der rein gerätebasierten Verfahren sind die satellitenbasierten Verfahren wie GPS.

Bei den netzwerkbasierten Verfahren, erfolgt die Positionsbestimmung im Netzwerk, in dem die Daten verschiedener Basisstationen ausgewertet werden, welche Signale vom Endgerät empfangen. Bekanntes Beispiel ist die Mobilfunkortung, bei der im Netzwerk die Position der Mobilfunkzelle (Cell-ID-Verfahren) ermittelt wird, in welcher das mobile Endgerät registriert ist.

Küppers [Küp05] spricht im Zusammenhang dieser Klassifikation auch von gerätebasiert und netzwerkgestützter bzw. netzwerkbasiert und gerätegestützter Lokalisierung, denn prinzipiell ist eine Bestimmung nur in Zusammenarbeit mit dem Gegenpart möglich. Je nachdem, wie umfangreich die Unterstützung erfolgt, verschwimmen die Grenzen zwischen gerätebasiert und netzwerkbasiert. Beispielsweise kann auf dem Endgerät die Sammlung von WLAN-Signalen stattfinden, welche dann aber zu einem Server übertragen und dort zur Positionsbestimmung

verarbeitet werden. Dies hat eher netzwerkorientierten Charakter. Eine andere Möglichkeit wäre die notwendigen Daten (Radiomap) aus dem Netzwerk auf das Endgerät zu übertragen und die Positionsberechnung lokal durchzuführen. Dies hat eher endgerätorientierten Charakter.

Bei der Bewertung eines konkreten Verfahrens ist daher darauf zu achten, ob die Bestimmung vollständig endgerätebasiert (also ohne Netzwerkunterstützung), gemischt, oder rein netzwerkbasiert (also ohne aktive Kooperation des Endgerätes) stattfinden kann. GPS zählt als rein endgerätebasiert. WLAN-Verfahren sind meist gemischt, aber auch rein netzwerkbasiert möglich. Mobilfunkortung (Cell-ID Verfahren) ist rein netzwerkbasiert (sofern sie von der Mobilfunkinfrastruktur aus erfolgt).

Durch Kombination von gerätebasierten und netzwerkbasierten Verfahren bildet sich eine dritte Gruppe, welche hybride Verfahren genannt wird. Ein Beispiel dafür ist Assisted GPS, bei dem zunächst per Mobilfunkortung der ungefähre Standort ermittelt wird (netzwerkbasiert) und anschließend als Grundlage zur GPS-Ortung (gerätebasiert) benutzt wird, die dadurch wesentlich schneller erfolgt, weil die zu erwartenden sichtbaren Satelliten bekannt sind.

3.2.2 Verfügbarkeit

Jede Ortungstechnologie hat bestimmte Voraussetzungen, die eine Nutzung überhaupt erst ermöglichen. Neben den technischen Eigenschaften des mobilen Endgerätes (z. B.: Existenz von GPS-Modul, WLAN-Schnittstelle, Mobilfunkschnittstelle, mobiler Internetzugang) spielt vor allem die Umgebung eine Rolle, in welcher der Endnutzer sich aufhält. Die Eigenschaften einer Ortungstechnologie, vor allem der erreichbaren Genauigkeit, können stark variieren und durch verschiedene Störfaktoren beeinflusst werden.

Indoor Indoor bedeutet innerhalb von Gebäuden. Wände und Decken aus Beton erschweren oder blockieren die Ausbreitung von Funksignalen.
Bis auf Bereiche in Fensternähe ist deswegen GPS nicht oder kaum anwendbar, da eine Sichtverbindung zu den Satelliten gegeben sein muss.
Gebäude sind häufig bereits flächendeckend mit WAPs ausgestattet. Daher ist WLAN eine der geeignetsten Ortungsmethoden. Fingerprintverfahren ermöglichen nach Einmessen der Umgebung, die Nutzung, auch ohne dass man an einem konkreten WAP eingewählt sein muss.
Bluetooth ist ebenfalls gut geeignet. Allerdings bedarf es hier zunächst eine Ausstattung der Umgebung mit Bluetoothbeacons.
Mobilfunkortung ist physikalisch gesehen möglich, der zu erreichende Genauigkeitsgrad erfüllt aber bei weitem nicht die Anforderungen für Anwendungen in Gebäuden. Oftmals kann kaum bestimmt werden ob man sich in einem bestimmten Gebäude befindet oder nicht, von der Bestimmung von Räumen oder Etagen ganz abgesehen.

Qualität	**Rural (ländlich)**	**Urban (städtisch)**	**Indoor**
GPS	sehr gut	mittelmäßig bis gut	schlecht oder garnicht
Mobilfunk	sehr schlecht	gut	gut
WLAN	nicht möglich	gut bis sehr gut	sehr gut
Bluetooth	nicht möglich	nicht möglich	sehr gut

Tab. 3.2: Zusammenhang Umgebung und erreichbare Qualität

Urban Urbane Gebiete, also innerstädtische Gebiete, sind gekennzeichnet durch hohen Bebauungsgrad, das heißt eine Vielzahl an Gebäuden, mit meist vielen Stockwerken. In diesen "Häuserschluchten" kann die Verwendung von GPS problematisch werden, da die Sichtverbindung zu ausreichend Satelliten nicht immer gegeben ist, oder die Signale durch Reflektionen an Gebäudefassaden zu verfälschten Berechnungen führen.
Die Verteilung von WLAN-Accesspoints ist sehr hoch, so dass auch außerhalb von Gebäuden zwischen Häuserblocks eine zufriedenstellende WLAN-Ortung möglich wird. Firmen wie Skyhook Wireless[19] oder Google haben bereits viele Städte eingemessen.
Mobilfunkortung ist sehr gut im urbanen Bereich anwendbar. Hohe Mobilfunknutzerzahlen haben die Aufstellung vieler Mobilfunkantennen erforderlich gemacht. Die Mobilfunkzellen sind dementsprechend klein, was sich in verbesserter Genauigkeit wiederspiegelt.

Outdoor Der Ländliche Bereich ist optimal für GPS geeignet, da es gute Sichtverhältnisse zu den Satelliten gibt. In bergigen Gebieten kann das Verhalten allerdings ähnlich wie in Häuserschluchten negativ beeinflusst werden.
Mobilfunkortung ist im ländlichen Bereich schlecht geeignet, da die Mobilfunkzellen sehr groß sind und die Ortungsgenauigkeit daher bis zu mehreren Kilometern betragen kann.
Bluetoth und WLAN scheiden auf Grund der kurzen Reichweite und mangelnder WAP oder Beacons ganz aus.

Wie man erkennen kann, stehen Umgebung und die erreichbare Qualität der Standortinformationen und prinzipielle Möglichkeit der Bestimmung in direktem Zusammenhang, wie in Abbildung 3.2 gegenübergestellt ist.

Aus dieser Eigenschaft lassen sich zwei Parameter ableiten: Leistung (*yield*) und Konsistenz (*consistency*). Die Leistung bewertet die Fähigkeit einer Ortungstechnologie in allen Umgebungen eine Standortbestimmung durchführen zu können. In Verbindung damit ist die Konsistenz das Maß für die Stabilität der Genauigkeit in allen Umgebungen. So hat Mobilfunk beispielsweise eine hohe Ausbeute, denn es funktioniert sowohl im ländlichen Bereich als auch indoor. Die Konsistenz ist allerdings sehr niedrig, da sich die Genauigkeit von ländlich über urban zu indoor

[19] http://www.skyhookwireless.com (Stand: 28.06.2010)

massiv unterscheidet. GPS hingegen hat eine mittlere Stabilität, da es sowohl im ländlichen wie städtischen Bereich funktioniert, indoor allerdings versagt. Die Konsistenz hingegen ist sehr hoch, da, wenn es funktioniert, recht gleichmäßige gute Genauigkeit erreicht wird.

Neben der grundlegenden technischen Voraussetzung einer konkreten Ortungstechnologie oder Lösung spielt meist auch das Vorhandensein der a-priori-Daten über Referenzpunkte oder Signalpatterns eine Rolle, welche für Positionsberechnungen benötigt werden. So muss bei WLAN-Verfahren zum Beispiel das Zielgebiet eingemessen (Radiomap) oder die Koordinaten der WAPs bekannt sein.

3.2.3 Qualität

Um Lokalisierungstechnologien vergleichbar zu machen, lassen sich aus deren Eigenschaften verschiedene Qualitätskriterien ableiten. Dazu zählt vor allem auch, welche Qualität der zu ermittelnden Standortinformationen erreicht werden kann, welche in Kapitel 2.4.2 erläutert wurden.

3.2.3.1 Genauigkeit

Einer der wichtigsten Qualitätsparameter ist die erreichbare Genauigkeit der zu ermittelnden Position. (Vergleich Kapitel 2.4.2)

3.2.3.2 Performanz

Performanz bezieht sich auf den bei der Standortbestimmung unvermeidlichen Overhead, sowohl beim Endgerät, der Infrastruktur und der Drahtlosschnittstelle. Die Netzbelastung wird auch als Signaloverhead bezeichnet und umfasst die Menge der Nachrichten die zwischen Endgerät und Infrastruktur, sowie innerhalb des Netzwerkes zur Ortungskontrolle ausgetauscht werden. Der Rechenaufwand wird Berechnungsoverhead genannt und bezieht sich auf die Rechenleistung in den Kontrolleinheiten/Datenbanken des Netzwerks und im Endgerät.
Performanz steht oft in Beziehung zur Genauigkeit. Ein hoher Grad an Genauigkeit verursacht einen hohen Overhead und umgekehrt.

3.2.3.3 Energieverbrauch

Mobile Endgeräte besitzen im Allgemeinen nur begrenzte Energieressourcen da sie durch Batterien oder Akkus betrieben werden. Ein hoher Verbrauch führt zu kürzeren Standby- und

Sprechzeiten und damit verringerter Nutzerakzeptanz. Deshalb muss sparsam mit diesen Ressourcen umgegangen werden. Je nach verwendeter Technologie und des anfallenden Signal- und Berechnungsoverhead fällt der Energieverbrauch zur Lokalisierung höher oder niedriger aus.

3.2.3.4 Latenz

Die Latenz (Verzögerung) bezieht sich auf die Zeitspanne zwischen der Positionsanfrage und Bereitstellung der ermittelten Position. Einfluss haben die zu erledigen Aufgaben wie zum Beispiel Auffinden und Auswählen ein oder mehrerer Basisstationen, Koordination des Positionierungsprozesses zwischen den beteiligten Komponenten, Zuweisung von Ressourcen, Durchführung von Messungen , Berechnung der Position auf Basis von Messergebnissen und Übertragung der ermittelten Daten.

Bei kontinuierlicher Ortung über einen längeren Zeitraum müssen die meisten Schritte nur einmal zu Beginn des Trackings durchgeführt werden. Ein wichtiger Indikator ist daher die Latenz der ersten ermittelten Position (position fix), auch “time to first fix” (TTFF) genannt. Die Größe der tolerierbaren TTFF hängt stark von Typ der LBS ab. Vor allem bei hochgradig interaktiven LBS wird ein hoher TTFF negativ erfahren und kann zu verringerter Attraktivität und Akzeptanz beim Anwender führen.

Wichtig sind also zwei Parameter: die Zeit bis zur ersten ermittelten Position (TTFF), und das minimale Zeitintervall für kontinuierliche Updates (Latenz).

3.2.3.5 Kosten

Installation und Betrieb von Ortungstechnologien verursachen Kosten. Einführungskosten umfassen die Installation der Infrastruktur wie Basisstationen, Datenbanken, Kontrolleinheiten oder die Erweiterung bestehender Infrastruktur, z. B.: Mobilfunknetzwerk mit Ortungsmöglichkeit zu erweitern. Betriebskosten beziehen sich auf die Unterhaltung der Technologien.
Aus Sicht des Nutzers und des LBS-Anbieters ist aber vor allem wichtig, welche konkreten Kosten für die Nutzung einer konkreten Ortungstechnologie anfallen. GPS zählt als eine preiswerte Lösung, da die Empfänger bereits in vielen Endgeräten verfügbar sind, und für den Endnutzer keine Kosten verursacht. Mobilfunkortung dagegen ist sehr preisintensiv, da jede Ortungsanfrage von den Mobilfunkbetreibern bzw. Dritten abgerechnet werden.

3.2.3.6 Privatsphäre

Die Standortdaten eines Nutzers sind sehr sensible und persönliche Daten welche das Risiko von Missbrauch tragen. Wichtige Voraussetzung für die Akzeptanz von LBS ist daher, dass der Nutzer

Kontrolle darüber hat wem, wann und in welcher Form seine Standortdaten verfügbar gemacht werden, und wie andere Akteure damit umgehen dürfen. Je nach Ortungstechnologie ist die Privatsphäre bereits mehr oder weniger gut geschützt. GPS hat eine sehr gute Privatsphäre, da es vollkommen gerätebasiert ohne weitere Netzwerkkommunikation auskommt. Mobilfunkortung hingegen wird vollständig netzwerkseitig durchgeführt, was viele Möglichkeiten eröffnet von Dritten ausgenutzt zu werden. Die Gesetzgebung hat deshalb strenge Vorschriften eingeführt[20].

3.3 Systeme

Unter Ausnutzung verschiedener Technologien und Infrastrukturen haben sich eine Reihe von Ortungssystemen herauskristallisiert, welche für die Nutzung mit mobilen Endgeräte gute geeignet sind und immer mehr Verbreitung finden. Die wichtigsten, konkret Mobilfunk, GPS, WLAN sollen im Folgenden genauer charakterisiert werden. Es ist besonders hervorzuheben, dass es für jede dieser Systeme wiederum verschiedene Variationen und Ansätze gibt (Ortungslösung), welche sich in den in Kapitel 3.2 genannten Eigenschaften unterscheiden. Eine annähernd genaue Spezifikation kann immer nur für eine konkrete Ortungslösung gegeben werden.

3.3.1 Mobilfunk

Mobilfunkortung wurde initiiert durch die E911 und E112 Mandate, welche gesetzlich festschreiben, dass bei Notrufen über Mobiltelefone der Anrufer bis auf eine geforderte Genauigkeit geortet werden können muss. Mobilfunkbetreiber mussten diese Forderung implementieren. Erste Verfahren nutzten dazu die Cell-ID der Mobilfunkstation über welche der Teilnehmer im Netz angemeldet ist. Dazu waren kaum Änderungen an der bisherigen Infrastruktur notwendig.

Die Positionsdaten werden von den Mopilfunkbetreibern im “Gateway Mobile Location Center” (GMLC) gegen Gebühr zugreifbar gemacht. Die Lokalisierung innerhalb der Mobilfunknetze wird durch systemspezifische Prozeduren kontrolliert.

Mobilfunkbetreiber außerhalb der USA vermeiden allerdings die Kosten zur Einführung fortgeschrittenerer Technologien und bieten nur Cell-ID Positionierung, welche oft nicht den Anforderungen aktueller LBS genügt [KTL06]

Die Genauigkeit für Cell-ID-Verfahren im innerstädtischen Bereich kann 50-500m (typischerweise 200m) betragen (Mikrozellen) oder 500m - 5 km (typischerweise 2km) (Makrozellen). In ländlichen Gebieten, wo die Zellen wesentlich größer sind kann die Genauigkeit 1-35km betragen.

[20] Bedingung laut §98 Abs. 1 TKG: schriftliche Genehmigung des Nutzers (Freischaltung via SMS für Ortungsdienstleister und bei einigen Netzbetreibern), nach einigen wenige Ortungen erfolgt Informierung des Nutzers per SMS dass er geortet wurde.

Verfahren	Genauigkeit	Grundlage	Erweiterung Netzwerk	Erweiterung Endgerät
Cell-ID	50-500m (urban, Mikrozelle) 500m-5km (urban, Makrozelle) 1-35km (rural)	Mobilfunkzelle	-	-
Angle of Arrival	300m	Winkel	Richtantennen Basistationen	-
Time of Arrival	125-200m	Laufzeit	LMU zur Zeitsynchronisation (nur bei GSM)	-
Time Difference of Arrival	125m	Differenzen Laufzeit	LMU zur Zeitsynchronisation (nur bei GSM)	-
Enhanced-Observed Time Difference	50-150m	wie Toa und TDoA, endgerätebasiert	-	Softwareerweiterung

Tab. 3.3: Eigenschaften der Verfahren für Mobilfunkortung

Unter Ausnutzung der Zeitslots bei der Mobilfunkübertragung kann das Cell-ID Verfahren verbessert werden. Das Verfahren wird dann Enhanced Cell-ID oder Cell-ID/TA genannt. Auf Grund von Mehrwegausbreitung der Signale und Reflektionen an Gebäuden kann das Ergebnis verfälscht werden, weshalb nur im ländlichen Gebiet eine Verbesserung erzielt werden kann. Statt der Zeitslots kann auch auf Signalstärkemessung zurückgegriffen werden, um den Abstand zur Basisstation der aktuellen Zelle zu bestimmen.

Weitere verbesserte Verfahren, die eine Erweiterung der Infrastruktur oder des Endgerätes voraussetzen sind das winkelbasierte Verfahren Angel of Arrival (AoA) und die zeitbasierten Verfahren Time of Arrival (TOA), Time Difference of Arrival (TDOA) und Enhanced-Observed Time Difference (E-OTD) (Vergleich Kapitel 3.1.2). Tabelle 3.3 stellt die Verfahren zur Mobilfunkortung im Überblick dar (basierend auf [BL09, Küp05]).

Der Vorteil von Mobilfunkortung ist, dass sie auf jedem über eine Mobilfunkschnittstelle wie GSM oder UMTS verfügenden Endgerät nutzbar ist. Die erreichbare Genauigkeit genügt allerdings nicht für feingranulare Anwendungen, wie man sie vor allem im Innenraumbereich vorfindet. Für LBS-Anbieter oder Endnutzer können Kosten entstehen, was einen weiteren Nachteil darstellt. Außerdem sind aus Datenschutzgründen aufwendige Genehmigungen[21] des Nutzers notwendig.

[21] vergleich §98 Abs. 1 TKG

Neben den integrierten Lösungen der Mobilfunkbetreiber, gibt es aber auch Alternativen von Fremdanbietern, welche sich das Mobilfunknetz zu Nutze machen. Datenbanken mit Informationen zu Mobilfunkzellen und deren Standort gibt es zum Beispiel von OpenCellID[22]. Durch Auslesen der Zelle auf dem Endgerät und Prüfen gegen die Datenbank lässt sich die ungefähre Position ermitteln. Andere Lösungen nutzen neben WLAN auch die Signalstärkeparameter des Mobilfunknetzes für das Fingerprintverfahren.

3.3.2 GPS

Das Globale Positioning System GPS ist das zurzeit verbreitetste satellitenbasierte Ortungssystem. Es wurde ursprünglich vom US-Militär entwickelt, später aber auch für den zivilen Gebrauch freigegeben. Es wird in zwei Services mit unterschiedlichen Eigenschaften unterschieden: Standard Positioning Service (SPS) für den zivilen Bereich und Precise Positioning Service (PPS) für den militärischen Bereich. Ursprünglich war SPS mit dem Feature “Selective Availability” (SA) ausgestattet, welches es ermöglicht die Genauigkeit zu begrenzen. SA wurde inzwischen allerdings deaktiviert.

Im zivielen Bereich ist damit eine typische Genauigkeit von bis zu 15m erreichbar (SPS ohne SA). Vorteile von GPS sind die mögliche Echtzeitortung, inklusive Bestimmung der Geschwindigkeit, die hohe Skalierbarkeit (unbegrenzte Anzahl von Nutzern), der hohe Verbreitungsgrad (bereits viele mobile Endgeräte sind mit GPS ausgestattet) und die hohe Genauigkeit.

Nachteil ist der hohe Energieverbrauch, aber vor allem die hohe TTFF. Es kann durchaus 30 bis 60 Sekunden oder länger dauern, bis die erste Position bestimmt ist, je nachdem wie aktuell die GPS-Daten sind, bzw. wann die letzte GPS-Standortbestimmung zurück liegt.

Um die Zeit der Standortbestimmung zu verkürzen kann der GPS-Empfänger vorab mit Daten versorgt werden, die er normalerweise über das Satellitensignal empfangen würde. Dazu stehen verschiedene Möglichkeiten zur Verfügung welche zusammenfassend als Assisted-GPS (A-GPS, unterstütztes GPS”) bezeichnet werden.[JSH07] Typisch ist zum Beispiel, per Mobilfunkortung eine ungefähre Position zu ermitteln und damit den Suchraum sichtbarer Satelliten einzuschränken. Eine andere Möglichkeit ist der Empfang von Korrekturdaten über eine Netzwerkverbindung von nahegelegenen stationären GPS-Empfängern, welche ungehinderte Sicht zu Satelliten haben. Dies ist zum Beispiel innerhalb von Gebäuden sinnvoll.

Weitere Verbesserungen und Genauigkeitsschwankungen können durch Metainformationen ausgeglichen werden wie z.B. Straßenverläufe bei Navigationslösungen.

[22] http://www.opencellid.org (Stand: 29.06.2010)

3.3.3 WLAN

WLAN-Ortung lässt sich überall dort einsetzen, wo es eine hohe Abdeckung mit WAPs gibt. In urbanen Gebieten und vielen Indoorbereichen wie Bürogebäude, Einkaufspassagen, Messen, ist das zunehmend der Fall.

Es lassen sich viele verschiedene Bestimmungsmethoden einsetzen, von Lateration basierend auf Signalstärke- und Laufzeitmessungen, Fingerprinting oder Kombinationen aus beidem. Zu den notwendigen a priori-Daten gehören die Radiomapdatenbanken, Koordinaten der WAPs oder weitere Infrastrukturdaten wie die Lage von Zimmer- und Gebäudewänden. Bestimmungen sind sowohl endgerätebasiert, oder netzwerkbasiert möglich.

Firmen wie Skyhook Wireless[23] oder Google[24] haben bereits viele Städte auf WLAN-Fingerprints ausgemessen und in Datenbanken abgelegt. Durch fortwährende Kallibrierung (zum Beispiel durch Rückmeldung von Nutzern die mit alternativen wie GPS ausgestattet sind) verbessert sich die Datenbank ständig selbst, ohne ein erneutes Einmessen zu fordern.

Auf Grund der vielfältigen Implementierungsmöglichkeiten wurden viele verschiedene Lösungen entwickelt, sowohl im Open-Source-Bereich z. B.: MagicMap[25] und Place Lab[26] als auch im kommerziellen Bereich beispielsweise Skyhook Wireless' XPS und Spotigos HyPS[27]
Problematisch wird das Fingerprintingverfahren für Szenarien, wie zum Beispiel Messen, bei denen sich die Aufstellung der WAPs jedesmal grundlegend neu zusammensetzt. Dort sind vorherige Einmessungen unumgänglich.

Die tatsächlich erreichbare Genauigkeit hängt von der konkreten Implementation und den verwendeter Verfahren ab, ist aber prinzipiell bis in den einstelligen Meterbereich möglich.

Vorteile der WLAN-Ortung sind die hohe Genauigkeit, die niedrige Latenz, und die hohe Verbreitung, sowohl auf Endgeräten als auch der Infrastruktur.

Die Nachteile liegen im hohen Energieverbrauch und dem oft notwendigen Einmessen. Kosten können entstehen wenn kommerzielle Anbieter und deren gepflegte Datenbanken genutzt werden sollen. Eine Garantie für vollständige Abdeckung gibt es allerdings nicht.

[KJ06]

[23] http://www.skyhookwireless.com/ (Stand: 29.06.2010)
[24] http://www.google.de/ (Stand: 29.06.2010)
[25] http://www.magicmap.de/ (Stand: 29.06.2010)
[26] http://www.placelab.org/ (Stand: 29.06.2010)
[27] http://www.spotigo.com (Stand: 20.10.2009)

3.4 LBS-Middleware

Nach dem Middleware Resource Center[28] bezeichnet man Middleware als *"any software that allows other software to interact"*. Middleware ist also eine Art Vermittler oder Zwischenschicht. Sie bietet gewöhnlich eine Reihe von (standardisierten) APIs, Protokollen und Diensten die von verschiedenen anderen Anwendungen genutzt werden können. Damit werden die Heterogenitäten der miteinander verbundenen Instanzen voreinander versteckt und die Entwicklung und Ausführung von komplexen und verteilten Anwendungen wesentlich vereinfacht.

Im Fall einer LBS-Middleware werden APIs, Protokolle und Dienste für LBS-Anwendungen angeboten. Der Umfang einer solchen Middleware kann dabei stark variieren und im besten Fall die komplette LBS-Beschaffungskette abdecken, vom Endgerät, über den Positionsbestimmer, Location Provider bis zu LBS- und Contentprovider [Küp05]. Damit werden die verschiedenen Infrastrukturen und ihre benutzten Protokolle vereint und ihre Heterogenität vor der jeweiligen LBS-Anwendung versteckt.

Im Fokus dieser Arbeit steht vor allem die dynamische und intelligente Auswahl konkreter, verfügbarer und vornehmlich endgerätezentrierter Ortungslösungen. Diese soll nicht von einzelnen LBS-Anwendungen sondern von einer Middlewarekomponente durchgeführt werden. Die Mechanismen, um all die verschiedene Aspekte der Standortbestimmung vor der LBS-Anwendung zu verstecken (Auswahl der Ortungsverfahren, Verteilung der Positionsdaten, Transformation in andere Formate, ggf. Kombinierung mit geografischem Kontext) werden als Ortungstransparenz bezeichnet. Die zu entwickelnde Lösung fokussiert also nur auf die unterste Schicht.

Kapitel 4 wird näher auf die Anforderungen an eine solche Ortungsmiddleware eingehen und aktuelle Lösungen aus der Forschung und dem kommerziellen Bereich analysieren.

[28] http://www.middleware.org/whatis.html (Stand: 28.06.2010)

4 Analyse und State of the Art

In den beiden vorangegangenen Kapiteln wurde auf die Grundlagen und Charakteristiken von LBS und der zu ihrer Ausführung notwendigen automatisierten Bestimmung des Standorts eingegangen. Es hat sich gezeigt, dass die Standortbestimmung ein sehr komplexes Gebiet ist, mit einer Vielzahl möglicher Technologien, Verfahren, Lösungen und damit auch verschiedener Eigenschaften und Möglichkeiten (Vergleich Abbildung 4.1).

Grundlage jeder automatisierten Standortbestimmung ist die Ausnutzung einer oder mehrerer Technologien. Dazu zählen die verschiedenen Drahtlosschnittstellen wie WLAN, Bluetooth, oder Mobilfunk, Satellitensysteme wie GPS oder Galileo oder auch mechanische Sensoren. Vor allem die Parameter Laufzeit, Richtung und Stärke der Funksignale spielen dabei eine große Rolle.

Die Ermittlung der Standortinformationen erfolgt schließlich durch Verfahren wie Nachbarschaftserkennung, Angulation, Lateration, Mustervergleich oder Koppelnavigation (Vergleich Kapitel 3.1).

Welche Verfahren mit welchen Technologien wie miteinander kombiniert werden, ist durch die jeweilige Ortungslösung (Ortungsdienst) festgelegt. Aus dem breiten Spektrum an Möglichkeiten wurden in den letzten Jahren viele verschiedene, unterschiedliche, sowohl kommerzielle als auch wissenschaftliche Ansätze entwickelt. So vielfältig und unterschiedlich die Ortungslösungen sind, so unterschiedliche Eigenschaften weisen sie auf.

Dazu zählt sowohl, welche Standortinformationen in welchem Format ermittelt werden können, z. B. geografische Koordinaten, symbolische Position oder Geschwindigkeit und Richtung der Bewegung, als auch mit welcher Qualität, z. B.: Genauigkeit, Latenz, Zuverlässigkeit, Aktualität, Kosten, Aufwand, die Bestimmung möglich ist.

Wie in der Einführung bereits ausgeführt wurde, ist es für LBS problematisch mit dieser Vielzahl unterschiedlicher Ortungsdienste umzugehen. LBS wissen nicht, welche Ortungsdienste verfügbar sind und welche am besten geeignet sind. Hinzu kommt, dass mehrere LBS mit unterschiedlichen Anforderungen gleichzeitig Standortdaten benötigen können.

Da das Management verschiedener Ortungsdienste nicht zu den Aufgaben von LBS zählt, sollte es von einer weiteren Instanz, einer Middleware durchgeführt werden. Das folgende Kapitel widmet sich der Anforderungsanalyse für eine Ortungsmiddleware. Es werden grundlegende Probleme identifiziert und erklärt, die bei Suche, Auswahl, Ausführung und Kombination verschiedener

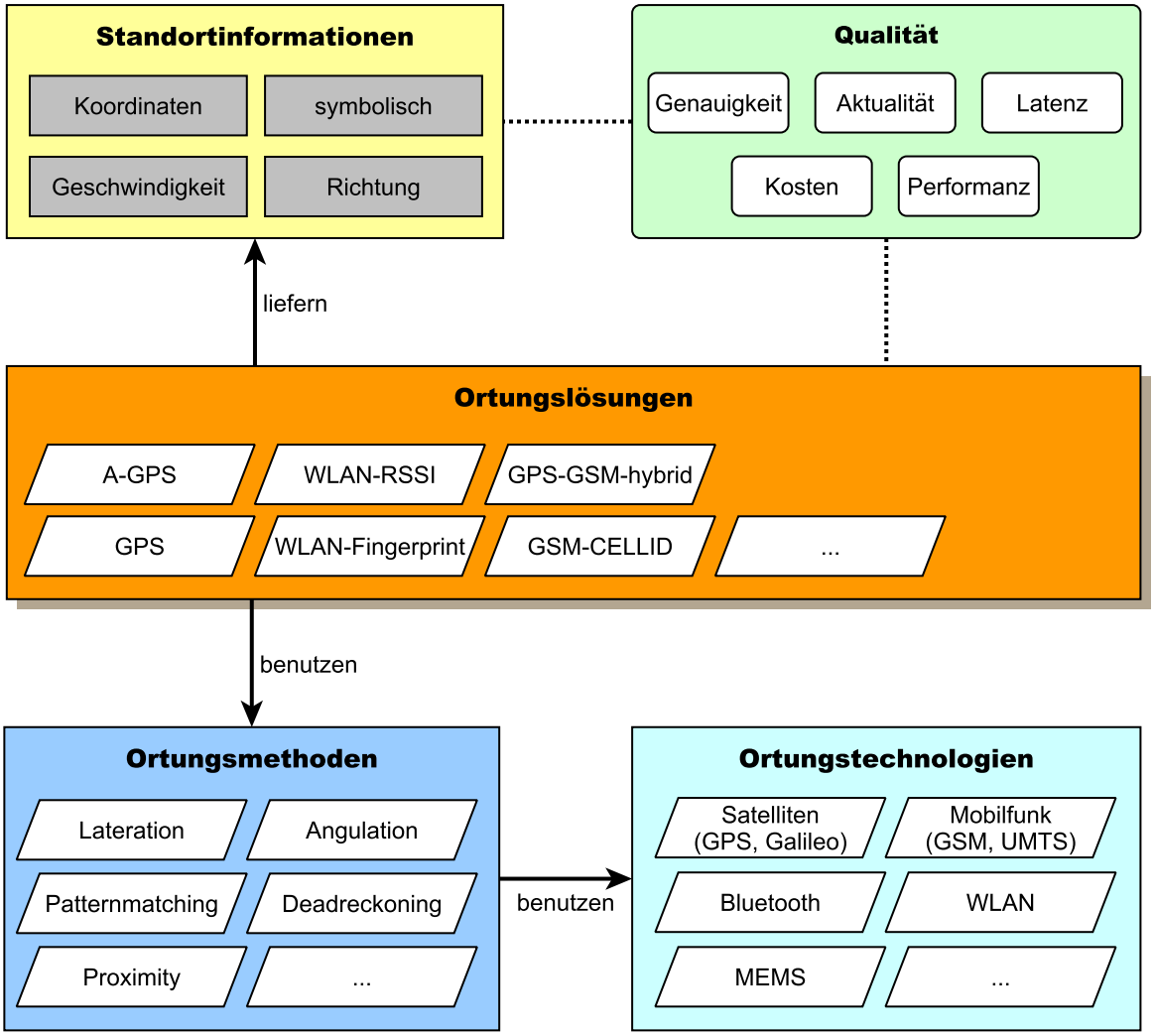


Abb. 4.1: Übersicht Standortbestimmung: Technologien, Methoden, Lösungen

Ortungsdienste auftreten können. Außerdem wird analysiert, welche wissenschaftlichen, kommerziellen und standardisierten Lösungsansätze es bereits gibt, und welche Schwächen oder Potentiale sie aufweisen.

4.1 Szenario

Zunächst werden zwei typische LBS Beispielanwendungen vorgestellt, aus denen dann verschiedene Anwendungsfälle für die Ortungsmiddleware abgeleitet werden. Anschließend werden die Probleme identifiziert, die beim Entwurf einer Ortungsmiddleware berücksichtigt werden müssen.

4.1.1 Beispielanwendungen

4.1.1.1 Wetterdienst

Ein Wetterdienst versorgt den Nutzer mit lokalen Wetterinformationen. Durch die automatische Lokalisierung des Nutzers sind die gelieferten Wetterdaten immer auf den aktuellen Standort des Nutzers zugeschnitten.

Die Anforderungen des Wetterdienstes an die Standortbestimmung sind gekennzeichnet durch niedrige Genauigkeit (1000m und mehr), kurze Antwortzeit (wenige Sekunden), niedrige Aktualität (mehrere Minuten) und einmalige Ortung. Wetterinformationen beziehen sich auf ein relativ großes Gebiet z. B. eine Stadt. Die benötigten Standortdaten müssen also keine hohe Genauigkeit aufweisen. Es handelt sich um eine einmalige Ortungsanfrage. Die Antwortzeit der Ortung sollte niedrig ausfallen, um eine hohe Wartezeit für den Nutzer zu verhindern. Die Aktualität der Standortdaten muss ebenfalls nicht sehr hoch ausfallen, da durch die grobe Genauigkeit nicht davon auszugehen ist, dassder Nutzer große Standortwechsel vollzogen hat.

Den Anforderungen entsprechend wäre eine typische Lösung Mobilfunkortung.

4.1.1.2 Fußgängernavigation

Ein Navigationsdienst für Fußgänger unterstützt den Nutzer bei seiner Tour durch die Einkaufsmeile einer Stadt und der Suche nach bestimmten Geschäften. Zunächst wird automatisch die aktuelle Position des Nutzers bestimmt, eine Route zur Position des gewünschten Geschäftes ermittelt. Um den Nutzer auf seinem Weg zum Geschäft navigieren zu können muss dessen Position kontinuierlich weiter verfolgt werden.

Die Anforderungen des Navigationsdienstes sind gekennzeichnet durch hohe Genauigkeit, mittlere bis kurze Antwortzeit, hohe Aktualität und kontinuierliche Ortung. Zur korrekten Fußgängernavigation, sowohl außerhalb als auch innerhalb von Gebäuden ist eine hohe Genauigkeit bis

auf wenige Meter notwendig. Da es sich bei Navigation um eine Echtzeitanwendung handelt müssen die Positionsdaten hochgradig aktuell und mit geringster Verzögerung geliefert werden. Die Latenz für die initiale, erste Position (TTFF) kann durchaus etwas höher liegen, ohne die Nutzerakzeptanz zu senken.

Typische Lösungen dafür sind GPS, welches allerdings nur außerhalb von Gebäuden zuverlässig funktioniert, oder WLAN-Ortung.

4.1.2 Schritte

Im Folgenden sollen die einzelnen Schritte aufgezählt werden, welche für die Ausführung von LBS unter Nutzung einer Ortungsmiddleware (Locationmiddleware LMW) durchzuführen sind. (Siehe Abb.: 4.2)

1. **Anfrage LBS** Der Nutzer stellt eine Anfrage an einen LBS.
2. **Anfrage Lokalisierung** Der LBS stellt eine Anfrage zur Lokalisierung, an einen konkreten Ortungsdienst oder wie in der Arbeit vorgesehen an eine Ortungsmiddleware.
3. **Lokalisierung** Der jeweilige Ortungsdienst führt die automatischen Standortbestimmung des Nutzers (bzw. seines Endgerätes) durch. Unter Verwendung einer Ortungsmiddleware sind weitere Unterschritte notwendig:

   Erkennung Je nach Voraussetzungen auf dem Endgerät des Nutzers, der Infrastruktur und weiterer Bedingungen werden die verfügbaren, nutzbaren Ortungsdienste ermittelt.

   Auswahl Ein oder ggf. mehrere dieser verfügbaren Ortungsdienste werden ausgewählt, in dem LBS-Anforderungen und Eigenschaften der verfügbaren Ortungsdienste abgeglichen werden.

   Ausführung Die Lokalisierung wird mit den gewählten Ortungsdiensten durchgeführt.

   Kombination Die ermittelten Standortdaten der jeweiligen Ortungsdienste werden zusammengeführt und ausgewertet. Dazu zählt umwandeln der Formate oder Datenfusion zur Qualitätsverbesserung.

   Überwachung Vor allem bei kontinuierlicher Ortung wird die gelieferte Qualität in festen Intervallen überwacht um die geforderten Anforderungen erfüllen zu können. Bei Abfall unter eine Grenze muss auf einen anderen Ortungsdienst gewechselt werden.
4. **Antwort Standort** Die lokalisierende Instanz gibt dir ermittelten Standortdaten und Qualitätseigenschaften an den LBS zurück.
5. **Generierung LBS-Daten** Der LBS filtert Inhaltsdaten auf Basis der ermittelten Standortdaten.
6. **Antwort LBS** Der LBS übermittelt die gefilterten Daten zur Darstellung beim Nutzer.

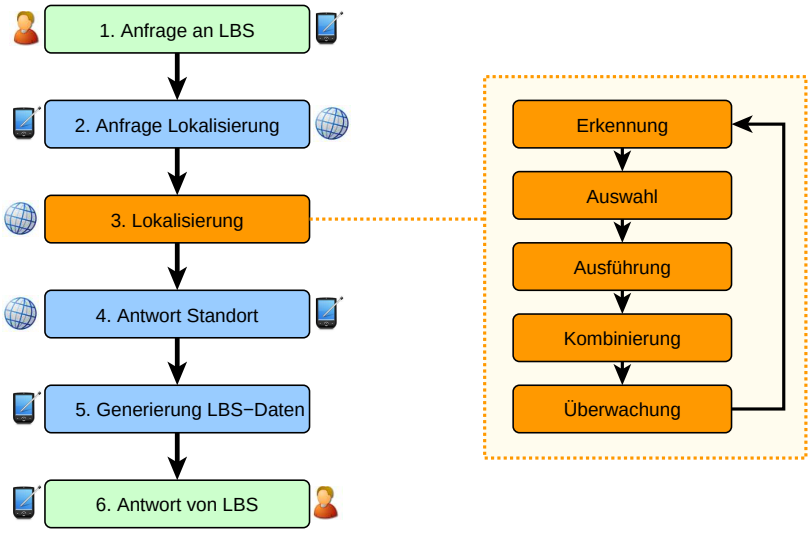


Abb. 4.2: Schritte der LBS- und Lokalisierungsausführung

Im Fokus dieser Arbeit stehen vor allem die Schritte zwei, drei und vier. Für die Ortungsmiddleware wird eine API benötigt, über welche LBS Anfragen stellen und Anforderungen spezifizieren können (2.) und über welche sie die ermittelten Standort- und Qualitätsdaten geliefert bekommen (4.). Durchführung der Lokalisation mit allen notwendigen Unterschritten ist Kern der Ortungsmiddleware (3.).

4.1.3 Anwendungsfälle

4.1.3.1 Auswahl

Ein wichtiger Anwendungsfall wurde mit Aufzählung der Schritte bereits genannt. Aus der Menge verfügbarer Ortungsdienste müssen ein oder ggf. auch mehrere Ortungsdienste ausgewählt werden. Dazu werden die Anforderungen des LBS mit den Eigenschaften der Ortungsdienste abgeglichen.

4.1.3.2 Nahtloser Wechsel

Durch verschiedene Ursachen wie z. B. Deaktivierung von Gerätefunktionen durch den Nutzer, Änderungen in der Umgebung kann sich die Verfügbarkeit der Ortungsdienste und die Qualität der Standortinformationen ändern. Neue Dienste können verfügbar werden, genutzte Dienste

können in Qualität nachlassen oder nicht verfügbar werden, nicht genutzte Dienste können bessere Qualität liefern.

Um vor allem bei kontinuierlicher Ortung, wie sie für Navigation oder Push-Dienste benötigt wird, fortwährend angemessene Standortdaten liefern zu können, muss die Nutzung konkreter Ortungsdienste dynamisch gesteuert werden.

Die tatsächliche Qualität der genutzten Ortungsdienste muss überwacht werden, um bei einem Abfall unter eine Grenze einen Wechsel durchführen zu können. Außerdem muss erfasst werden, ob alternative Ortungsdienste verfügbar geworden sind, welche möglicherweise bessere Ergebnisse liefern können.

Ist bei der Fußgängernavigation zum Beispiel GPS gewählt und aktiv, kann ein Wechsel aus verschiedenen Gründen notwendig sein. Der Nutzer könnte das GPS-Modul manuell deaktivieren, oder der Nutzer bewegt sich von draußen in das Innere eines Gebäudes, so dass der Sichtkontakt zu den Satelliten abbricht und keine oder nur noch sehr ungenaue Standortdaten geliefert werden. Die Middleware muss dieses Ereignis erkennen und auf einen alternativen Ortungsdienst wechseln, z. B.: eine lokal verfügbare WLAN-Ortung.

Der nahtlose Wechsel von Ortungsdiensten sollte vollständig versteckt vor nutzenden LBS erfolgen.

4.1.3.3 Mehrere parallele Ortungsanfragen

Es kann vorkommen, dass (nahezu) zeitgleich, verschiedene LBS ausgeführt werden und deshalb auch verschiedene parallele Ortungsanfragen, mit durchaus verschiedenen qualitativen Anforderungen gestellt werden. Jede Anfrage könnte einzeln betrachtet zu einer anderen Reaktion der Ortungsmiddleware führen, was sich in unterschiedlicher Auswahl der Ortungsdienste und damit Standortdaten unterschiedlicher Qualität äußern würde.

Der angesprochene Wetterdienst könnte zum Aufruf einer Mobilfunkortung führen, während die Fußgängernavigation mit GPS-Daten versorgt wird. Werden jetzt beide LBS-Anwendungen parallel verwendet, zum Beispiel weil der Nutzer während der Navigation das Wetter abrufen will, könnten im ungünstigen, klassischen Ansatz zwei verschiedene Ortungslösungen ausgeführt werden, was eine Verschwendung von Kosten und Ressourcen darstellt.

Die Herausforderung für die Ortungsmiddleware besteht darin, die Standortermittlung so auszuführen, dass die Anforderungen aller anfragenden LBS erfüllt werden, dabei aber der geringstmögliche Aufwand betrieben wird. Ausschlaggebend ist bei einer solchen *Requestaggregation* (Anforderungszusammenfassung) dabei die LBS mit den höchsten Anforderungen. Im konkreten Beispiel würde der Wetterdienst eine Antwort mit Standortdaten erhalten, welche auf den für den Navigationsdienst ermittelten GPS-Daten basiert. Dieses Verhalten bezeichnet

4.1.3.4 Transformation

Nicht jeder Ortungsdienst liefert Standortdaten in allen Formaten. Manche liefern geografische Daten in Form von Koordinaten, manche liefern symbolische Daten entsprechend eines bestimmten Modells oder Ontologie, manche liefern Bewegungsdaten wie Richtung und Geschwindigkeit.

Manche Formate lassen sich gut in andere überführen. Zum Beispiel lässt sich aus dem symbolischen Standort eines Raumes in einem Gebäude oft passende Koordinaten ermitteln. Aus Koordinaten wiederum lässt sich die nächstgelegene Adresse ermitteln. Aus kontinuierlich gemessenen Koordinaten kann man Angaben über Richtung und Geschwindigkeit ableiten.

Je nach dem welches Format von der LBS angefragt wird müssen entweder Ortungsdienste ausgewählt werden, welche das gewünschte Format direkt liefern oder welche, deren Ergebnisse durch eine Transformation in das gewünschte Format überführt werden können.

4.1.3.5 Kombination

Werden mehrere Ortungsdienste gleichzeitig verwendet, müssen die Ergebnisse zusammengeführt werden, um eine Antwort an die anfragenden LBS senden zu können.

Komplementäre Daten können einfach zu gehaltvolleren Standortdaten ergänzt werden, zum Beispiel Koordinaten, Richtung und Geschwindigkeit eines GPS-Moduls und symbolische Informationen eines WLAN-Ortungsdienstes.

Liegen gleichartige Daten vor, bieten sich mehrere Möglichkeiten. Entweder es wird eine Auswahl getroffen, oder die Daten werden verschmolzen (fusioniert). Die enthaltene Redundanz kann dabei möglicherweise zur Steigerung der Qualität genutzt werden. Je nachdem, welches Format die Daten haben gibt es vielfältige Möglichkeiten dieses verschmelzen durchzuführen. Koordinaten verschiedener Ortungsdienste könnten zum Beispiel durch Mittelwertbildung zu neuen Koordinaten zusammengefasst werden.

Grundlegende Voraussetzung zur Verschmelzung ist ein gemeinsames Referenzsystem. Die Daten verschiedener Quellen müssen also u. U. zunächst transformiert werden.

4.2 Schlüsselfragen und Probleme

Im Folgenden sollen wichtige Schlüsselfragen gestellt werden, welche Probleme aufzeigen, die bei der Konzeption der Middleware auftreten können. Die gewonnenen Erkenntnisse sind wichtig, zur Ableitung von Anforderungen an die Middleware und Analyse bisheriger Arbeiten auf dem Gebiet.

4.2.1 Bewertung von Ortungsdiensten

Um eine Auswahl treffen zu können, welcher Ortungsdienst für eine Anfrage geeignet ist und benutzt werden soll, müssen seine Eigenschaften bekannt sein. Es müssen also vorab verschiedene Kriterien bewertet werden, welche dann mit den Anforderungen der LBS-Anwendung abgeglichen werden.

4.2.1.1 Was wird bewertet?

Als Kriterien zur Bewertung können die in Kapitel 3.2.3 vorgestellten Qualitätsparameter (Genauigkeit, Latenz, Performanz, Energieverbrauch, Kosten, Datenschutz) dienen. Außerdem spielt der Datentyp eine Rolle, welche der Ortungsdienst liefern kann, zum Beispiel Koordinaten, Symbolische Informationen, Bewegungsdaten.

4.2.1.2 Auf welcher Grundlage wird bewertet?

Wichtig ist auch, auf welcher Grundlage die Bewertung vorgenommen wird und welche Aussagekraft dahintersteht. Vor allem bei Verfahren mit schlechter Konsistenz, schwanken die zu bewertenden Eigenschaften sehr stark in Abhängigkeit der Umgebung. So ist zum Beispiel die Genauigkeit von Mobilfunkortung (Cell-ID) innerstädtisch im besten Fall bis 200m, auf dem Land aber bis zu mehreren Kilometern. Ähnlich verhält es sich bei GPS. Ist man outdoor unterwegs lässt sich eine Genauigkeit bis zu 5 Metern erreichen, sobald man aber ein Gebäude betritt nimmt die Genauigkeit auf Grund der eingeschränkten Sicht zu Satelliten drastisch ab.

Solche Eigenschaften sind sehr dynamisch und hängen von der aktuellen Situation und Position des Nutzers ab. Eine genaue Angabe ist somit nicht möglich. Vielmehr lässt sich nur ein Richtwert angeben, welcher den typischen, günstigsten Fall wiederspiegelt. Solch eine unscharfe Angabe kann als a priori-Bewertung für die Auswahl benutzt werden. Welcher Wert aber tatsächlich zutrifft lässt sich erst feststellen, nachdem der Ortungsdienst ausgeführt wird und echt Werte ermittelt wurden.

Die Ermittlung dieser Richtwerte muss im Vorfeld und über einen längeren Zeitraum erfolgen, zum Beispiel durch den Anbieter des Ortungsdienstes. Es ist auch vorstellbar, dass während der Nutzung Statistiken gesammelt werden, und so eine Kalibrierung ermöglicht wird, die auf das Verhalten des jeweiligen Nutzers angepasst ist.

4.2.1.3 Wie wird bewertet?

Quantifizierung der Bewertungskriterien und Spezifikation der Anforderungen durch LBS kann auf verschiedene Arten erfolgen. Es können zu den Kriterien passende Einheiten gewählt werden,

wie zum Beispiel Genauigkeit in Meter, Intervall in Millisekunden. Es ist auch denkbar allgemeine Werte einer Skala zu vergeben, z. B.: von 0 bis 10 und bei der Bewertung aus den Eigenschaften den treffenden Wert abzuleiten.

4.2.2 Auswahlproblem

Um eine Abschätzung treffen zu können, welcher Ortungsdienst zu einem bestimmten Zeitpunkt zu benutzen ist müssen verschiedene Informationen einbezogen werden:

- Qualitätsanforderungen des LBS
- Qualitätseigenschaften der verfügbaren Ortungsdienste
- Restriktionen und Richtlinien (zum Beispiel aus Profileinstellungen des Nutzers)
- Ermittelte Echtzeitdaten der Ortungsdienste

Das Problem der Bewertung von Ortungsdiensten führt weiter zum Problem der Auswahl eines oder mehrerer Ortungsdienste. Wie kann man die tatsächlich lieferbare Qualität eines Ortungsdienstes feststellen, ohne ihn tatsächlich auszuführen? Oder wie kann man die Einsatzfähigkeit eines Ortungsdienstes feststellen, bevor man den Standort des Nutzers ermittelt hat?

Verschiedene Strategien sind denkbar. Man könnte alle potentiellen Ortungsdienste parallel auszuführen, und die besten Ergebnisse verwerten. Oder man sortiert die verfügbaren Ortungsdienste nach dem theoretischen Potential und führt einen nach dem anderen aus, bis die gewünschte Qualität erreicht wird oder keine weitere Verbesserung möglich ist.

Welche konkrete Auswahlstrategien und welche Algorithmen angewendet werden, ist eine Kernfrage für den Entwurf einer Ortungsmiddleware und führt weiter zum Problem der grundlegenden Strategie.

4.2.3 Grundlegende Strategie

Bei der Ausführung von LBS und Standortbestimmung treffen zwei gegensätzliche Interessen aufeinander. Der Nutzer will auf zuverlässige, zeitnahe Daten zugreifen, weshalb der LBS in den meisten Fällen die Anforderungen an best- und schnellstmögliche Standortbestimmung stellt: hohe Genauigkeit, Zuverlässigkeit und niedrige Latenz.

Dem entgegen steht der Aufwand für die Standortbestimmung. Energiebedarf, Rechenzeit, Kosten nehmen mit steigender Ortungsqualität gewöhnlich zu. Der Nutzer ist allerdings an minimalen Kosten und maximaler Akkudauer interessiert.

Das optimale Verhältnis zu finden ist nicht einfach, und meist muss priorisiert werden, was wichtiger ist: Aufwand oder Qualität.

Bei der Umsetzung einer Ortungsmiddleware sollten daher auch verschiedene Strategien integriert werden können. Unterschieden werden kann, ob einer, mehrere oder alle Ortungsdienste ausgewählt werden, oder ob nur der beste oder der energiesparendste Ortungsdienst ausgewählt wird.

4.2.4 Wechselmanagement

Ein weiterer wichtiger Punkt ist, wie das Management zum Wechseln von Ortungsdiensten erfolgt. Die Neubewertung der aktuell genutzten Ortungsdienste könnte nach einem festgelegten Zeitintervall, nach zurücklegen einer festen Entfernung des Nutzers oder bei jedem Eintreffen eines neuen Standortupdates erfolgen.

4.2.5 Handlungsspielraum der Ortungsmiddleware

Voraussetzung eines Ortungsdienstes kann das Aktivieren einer bestimmten Technologie oder Sensors sein, z. B. GPS-Modul, WLAN-Modul, Bluetooth-Modul. Eine wichtige Frage ist, wie und von wem diese Handlung durchgeführt werden darf: aktiv von der Middleware, aktiv durch den Nutzer oder indirekt über Profile und Richtlinien. Bei letzterem ist wiederum die Frage, durch wen diese ausgewählt werden, den Nutzer oder die LMW.

4.2.6 Wirtschaftliche Aspekte - Kosten

Aufstellung und Betrieb eines Ortungsdienstes sind mit Kosten verbunden weshalb die Nutzung u. U. kostenpflichtig sein kann. Dieser Aspekt muss in einer Ortungsmiddleware berücksichtigt werden.

Da ein LBS-Nutzer Ortungsdienste nur indirekt nutzt, fallen die Kosten zunächst auch nicht direkt an, sondern für den Nutzer des LCS, den LBS-Anbieter. Typischerweise besteht ein Nutzungsvertrag zwischen LBS-Anbieter und LCS-Anbieter, in dem z. B. definiert ist, dass pro Ortungsanfrage ein bestimmter Betrag abzurechnen ist. Der LBS-Anbieter muss selbst entscheiden wie er die Kosten deckt, z.B. durch Werbung, oder direkt beim LBS-Nutzer.

Eine solche enge Bindung zwischen LBS-Anbieter und LCS-Anbieter kompliziert den dynamischen Ansatz adaptiv nutzbarer LCS.

Eine denkbare Möglichkeit wäre zum Beispiel über Identitäten und Authentifizierungen festzustellen, ob ein spezieller LBS-Anbieter für die Nutzung eines LCS berechtigt ist, und damit das Angebot verfügbarer LCS zu erweitern oder einzuschränken.

Diese wirtschaftlichen Aspekte sind nicht trivial und nicht im Fokus dieser Arbeit, und werden daher nicht weiter betrachtet. Sie sollten aber in weitergehenden Arbeiten näher untersucht werden, da sie einen hohen praktischen Stellenwert haben.

4.2.7 Qualitätsbewusstsein

Im Gegensatz zum pragmatischen Ansatz des simplen Anforderns von Standortdaten, Ausnutzung der gelieferten Ergebnisse ohne weitere Beachtung der tatsächlichen Qualität steht der qualitätsbewusste Ansatz. Das heißt die LBS-Anwendung hat die Möglichkeit sich der Qualität der Standortdaten bewusst zu sein. Sie kann Qualitätsanforderungen spezifizieren. Die für die Lokalisierung verantwortliche Instanz wird dann versuchen diese Anforderungen zu erfüllen, die ermittelten Standort- und Qualitätsinformationen zurückliefern. Die LBS kann diese Daten verarbeiten und ggf. weitere Handlungen bei der LMW anfragen, je nachdem, ob eine zu niedrige Qualität ausreicht, oder weitere Ortungsdienste aktiviert werden sollen um eine Verbesserung zu erreichen. Wichtig in dem Zusammenhang ist sicher, dass bei Spezifikation der Anforderungen auch eine Gewichtung vorgenommen wird. So könnte eine Anfrage beinhalten dass eine bestimme Genauigkeit auf jeden Fall erfüllt werden soll, egal wie lange die Ermittlung dauert. In einer anderen Anfrage wäre die Latenz ein ausschlaggebendes Kriterium, d.h. innerhalb einer gewissen Zeitspanne muss ein Ergebnis vorliegen, bei zweitrangiger Genauigkeit.

4.2.8 Finden von Ortungsdiensten

Um einen modernen, flexiblen und erweiterbaren Ansatz zu unterstützten ist der Punkt des Suchen und Findens von Ortungsdiensten von großer Bedeutung. Es muss möglich sein, verschiedene Ortungsdienste auffinden zu können, welche für die aktuelle Ausführung von LBS verfügbar und nutzbar sind. Dazu zählen Ortungsdienste auf dem Endgerät des Nutzers (endgeräteorientiert) wie z. B.: GPS oder WLAN-Ortungsclient, sowie im Netzwerk (netzwerkorientiert) wie z. B.: Mobilfunkortung oder netzwerkbasierte WLAN-Ortung. Auch eine Betrachtung von globalen Ortungsdiensten (wie Mobilfunkortung oder GPS) oder lokalen Ortungsdiensten (wie eine WLAN-Ortung für eine Einkaufspassage) könnte wichtig sein.

Für einen solchen Ansatz müsste es ein Dienstverzeichnis geben, bei dem sich die verschiedenen Ortungsdienste registrieren. Die Ortungsmiddleware bezieht aus diesem Verzeichnis die verfügbaren Ortungsdienste zur Auswahl und Aktivierung.

4.3 Anforderungen

Ein Ziel der Arbeit besteht in der Konzeptionierung einer mobilen Ortungsmiddleware für mobile Endgeräte, welche die qualitätsbewusste, dynamische und flexible Ermittlung von Standortinformationen zur Ausführung mobiler LBS ermöglicht. Anschließend soll eine prototypischen Implementierung aufzeigen wie das Konzept umgesetzt werden kann und einige der Aspekte demonstrieren. Zusammenfassend werden die folgenden Anforderungen gestellt:

Anforderung	Beschreibung	Kriterium
Spezifikation Typ		
einmalig	einmalige Standortbestimmung	muss
periodisch	kontinuierliche Standortbestimmung in festgelegtem Intervall	muss
push	Registrierung für Pushevents, d. h. ein verwendeter Ortungsdienst schickt eine Nachricht falls ein vorgegebener Bereich betreten wird.	kann
Spezifikation Anforderungen		
Standortformat	Spezifikation der geforderten Standortformate, z. B. Koordinaten (Höhe, Breite), Symbolisch (Gebäude, Etage, Raum), Bewegungsdaten (Geschwindigkeit, Richtung)	muss
Qualität	Qualitative Anforderungen an die Bestimmung und Standortinformationen wie zum Beispiel Genauigkeit, Aktualität, TTFF	muss

Tab. 4.1: Anforderungen API Anfragen

Anforderung	Beschreibung	Kriterium
Rückgabe		
Standortdaten	die ermittelten Standortdaten, in den angeforderten Formaten	muss
Qualitätsangebot	das ermittelte, tatsächliche Qualitätsangebot der Standortdaten	muss
Status	Information über den Status der Anfragebearbeitung und verfügbarer Standortinformationen	kann

Tab. 4.2: Anforderungen API Rückgabe

4.3.1 Einheitliche API für LBS

Die Middleware benötigt eine einheitliche Programmierschnittstelle für LBS-Anwendungen um Standortanfragen entgegenzunehmen und Standortantworten zu senden. Tabelle 4.1 listet die Anforderungen an Standortanfragen und Tabelle 4.2 die Anfragen an die Standortrückgabe.

4.3.2 Dynamisches Ausführen von Ortungsdiensten

Den Kern der Middleware stellt das dynamische Suchen, Finden und Ausführen von Ortungsdiensten dar. Tabelle 4.3 listet die Anforderungen an die Ausführung der Standortbestimmung durch die Middleware.

Anforderung	Beschreibung	Kriterium
Dynamisches Ausführen von Ortungsdiensten		
Suchen und Finden	Ermittlung möglichere Ortungsdienste	
	auf dem Endgerät	muss
	in der Umgebung des Nutzers	kann
Auswahl	Auswahl einer oder mehrerer geeigneter Ortungsdienste durch Abgleichen der Anforderungen mit den Eigenschaften verfügbarer Dienste	muss
Ausführung	Ausführung der Standortbestimmung	muss
	Initiierung der ausgewählten Ortungsdienste	muss
	Sammlung der ermittelten Standortdaten und Qualitätsparameter	muss
	Transformation in andere Formate	kann
	Aggregation mehrerer Standortdaten	muss
Überwachung & Wechsel	Überwachen der Qualitätsparamter bei kontinuierlicher Ortung und Wechsel auf alternative Ortungsdienste	kann
Zwischenspeichern	Zwischenspeichern der zuletzt ermittelten Standortdaten	kann
Mehrere LBS	Bedienung mehrerer LBS-Anfragen auf einer gemeinsamen Basis, entsprechend den höchsten Anforderungen (Anfragenaggregation)	kann

Tab. 4.3: Anforderungen Dynamische Ausführung der Ortungsdienste

Anforderung	Beschreibung	Kriterium
Erweiterbarkeit		
Dienstsuche	Möglichkeit zum Entdecken und Einbinden neuer, für die Middleware bisher unbekannter Ortungsdienste	muss
Flexible Architektur	Flexible, geschichtete Architektur, um Komponenten austauschbar und erweiterbar zu machen, vor allem die Strategien und Methoden zur Datenfusion, Transformation, Auswahl und Wechsel.	muss

Tab. 4.4: Anforderungen Erweiterbarkeit

4.3.3 Erweiterbarkeit

Die Ortungsmiddleware soll flexibel und erweiterbar gestaltet werden, sowohl was die nutzbaren Ortungsdienste angeht, als auch die Verfahren und Methoden zur Auswahl und Anbindung der Ortungsdienste und Verarbeitung der Standortinformationen. Tabelle 4.4 listet die Anforderungen an die Erweiterbarkeit der Middleware.

4.4 State of the Art

Das folgende Unterkapitel soll einen Einblick in den aktuellen Stand der Technik im Bereich dynamischer Auswahl von Ortungsdiensten und Ortungsmiddlewareansätzen geben. Es sollen sowohl kommerzielle Entwicklungen, Standardisierungsversuche als auch wissenschaftliche Forschungsansätze untersucht werden.

4.4.1 Kommerzielle Systeme

Im kommerziellen Bereich gibt es viele Anbieter die sich auf die Bereitstellung von Ortungslösungen spezialisiert haben. Obwohl ssich anfangs nur auf einzelne Technologien konzentriert wurde, haben auch sie die Problematik der schwankenden Verfügbarkeit und Genauigkeit erkannt und sind dazu übergegangen, verschiedene Technologien zu kombinieren.

Einen der ersten Hybridansätze hat Skyhook Wireless vermarktet. Sie kombinieren WLAN, GPS und Mobilfunkortung (Funkturmtriangulierung) in einer hybriden Clientanwendung. Die WLAN-Ortung erfolgt durch das Fingerprintverfahren, weshalb die Leistung von Skyhook neben der Software zur Auswahl und Berechnung des Standorts vor allem auch in der kontinuierlichen Erweiterung einer weltweiten Radio-Map-Datenbank zählt. Damit sollen Genauigkeiten von 10-20 Meter, Nutzung innerhalb und außerhalb von Gebäuden und eine TTFF unter einer Sekunde[29].

Einen gleichen Ansatz verfolgt Spotigo[30][31] mit Hybrid Positioning Solution (HyPS), ebenfalls eine Kombination aus Mobilfunk, GPS und WLAN und dem Ziel einer weltweiten Radio-Map-Datenbank.

Vorteil solcher kommerzieller Systeme ist, dass viel Ressourcen verfügbar sind, um effiziente Algorithmen zu entwickeln, Infrastruktur zu verwalten und vor allem Daten zu sammeln, welche vor allem für WLAN-Ortung notwendig sind.

Der Nachteil sind allerdings die geschlossenen proprietären Systeme, welche nicht offen und mit alternativen Ortungsmöglichkeiten erweiterbar sind. Von außen ist die innere Funktionsweise nicht ersichtlich. Spezifikation von Anforderungen, auf deren Basis eine dynamische Bindung erfolgt ist bei den genannten Lösungen ebenso wenig möglich. Der Nutzer muss auf die Funktionsfähigkeit und Ausgewogenheit der Algorithmen vertrauen.

[29] http://www.skyhookwireless.com/flash/loader_howitworks.swf (Stand: 29.06.2010)
[30] http://www.spotigo.com (Stand: 20.10.2009)
[31] Laut Bekanntmachung des Amtsgerichts Düsseldorf vom 25.05.2010 (Aktenzeichen: HRB 53412) wurde die Firma Spotigo GmbH infolge der Eröffnung des Insolvenzverfahrens über ihr Vermögen aufgelöst.

4.4.2 Standardisierungen

4.4.2.1 JSR-179 Location API for J2ME

Für J2ME-basierte[32] Smartphones gibt es die Java Location API, spezifiziert unter JSR-179[33] Sie bietet eine standardisierte Schnittstelle zur Integration und Management darunterliegender Ortungsmethoden, sogenannter *LocationProvider*. Zur Anfrage einer Lokalisierungsmethode können verschiedene Kriterien definiert werden, welche von der Lokalisierungsmethode erfüllt werden müssen, z. B.: Genauigkeit, maximale Antwortzeit, erlaubter Energieverbrauch. Ist kein Provider existent der die Kriterien erfüllt wird *null* zurückgegeben [DG07].

Die Implementation erfolgt durch den Gerätehersteller. Er entscheidet damit, welche Lokalisierungsmethoden bereitgestellt werden. Eine Abfrage unterstützter Methoden, und welche Kriterien diese erfüllen müssen ist nicht möglich. Außerdem ist es ebenfalls nicht möglich neue Methoden zu ergänzen.

Das Management der Lokalisierungsmethoden ist darüberhinaus sehr unflexibel, da zur selben Zeit nur ein Provider auswählt werden kann. Außerdem erfolgt die Auswahl nur zum Instanziierungszeitpunkt. Die Qualität muss im weiteren Verlauf vom LBS selbst überwacht werden, um einen Qualitätsabfall zu erkennen [PB06].

Ein flexibles Datenschutzmodell gibt es ebenfalls nicht [KJ06].

4.4.2.2 GeoLocation API

Die Geolocation API[34] ist Teil der kommenden HTML5 Spezifikation[35] und bietet Webanwendungen eine High-Level-Schnittstelle zur Ermittlung des Nutzerstandortes. Die API ist vollkommen unabhängig von der konkret verwendeten Ortungsmethode. Dabei ist prinzipiell alles möglich: GPS, RFID, WiFi, Bluetooth, Mobilfunk, IP-Adresse oder manuelle Nutzereingaben. Es werden einmalige Standortanfragen und kontinuierliche Positionsaktualisierung unterstützt.

Die Implementierung erfolgt direkt im Browser. Dieser ist also maßgeblich für das Management verschiedener unterstützter Ortungsdienste verantwortlich. Mozilla Firefox[36] und Google Chrome[37] zum Beispiel nutzen sowohl GPS als auch WLAN-Ortung (unter Nutzung der Google Location Services) und IP-Geolocation als Fallbackvariante. Die API schlägt damit eine Brücke zwischen Webanwendungen und dem transparenten Zugriff auf Gerätefunktionen zur Standortbestimmung.

[32] Java Platform Micro Edition http://java.sun.com/javame/index.jsp (Stand: 29.06.2010)
[33] Java Specification Request 179 http://jcp.org/en/jsr/detail?id=179 (Stand: 29.06.2010)
[34] Geolocation API Specification Editor's Draft 10 February 2010 http://dev.w3.org/geo/api/spec-source.html (Stand: 29.06.2010)
[35] http://dev.w3.org/html5/spec/Overview.html (Stand: 29.06.2010)
[36] http://www.mozilla-europe.org/de/firefox/ (Stand: 29.06.2010)
[37] http://www.google.de/chrome (Stand: 29.06.2010)

Datenschutz wird sichergestellt indem der Nutzer bei jeder Anfrage erst zustimmen muss, ob seine Standortdaten an die jeweilige Webanwendung übermittelt werden dürfen.

Bewertung Obwohl HTML5 noch in Entwicklung ist, wird die Geolocation API bereits von vielen Browsern unterstützt. Sie ermöglicht es Webanwendungen auf einfache und standardisierte Weise Zugriff auf die Standortinformationen des Nutzers zu erhalten. Dabei wird endlich auch ohne proprietäre Clientsoftware die Nutzung von endgerätebasierten Ortungslösungen möglich, was das Potential für LBS-Webanwendungen deutlich erhöht. Ein gutes Sicherheitskonzept bietet die Voraussetzung für breite Nutzerakzeptanz.

Unter dem Fokus qualitätsbewusster, dynamischer Auswahl und Aggregation verschiedener Ortungsdienste gibt es aber auch eine Reihe von Nachteilen. Welche Ortungstechnologien jeweils konkret genutzt werden hängt nicht nur von den technologischen Eigenschaften des Endgerätes sondern auch vom verwendeten Browser und den implementierten Methoden ab. Die Webanwendung hat keinen Einfluss darauf, wie die Ermittlung erfolgt, was schließlich ausgewählt wird und erhält keine Garantien über Zuverlässigkeit und Genauigkeit der ermittelten Standortinformationen. Dynamik, Flexibilität und vor allem Qualitätsbewusstsein sind dadurch sehr eingeschränkt.

4.4.2.3 Android Location API

Android[38] ist das von der Open Handset Alliance[39] entwickelte quelloffene Betriebssystem für mobile Endgeräte. Mobile standort- und kartenbasierte Anwendungen sind ein wichtiger Punkt des Androidkonzeptes. Für den Zugriff auf Standortinformationen gibt es ein integriertes *location framework*.

Die zentrale Komponente ist der *LocationManager*, welche die API bietet, zur Bestimmung von Standort und Ausrichtung/Orientierung des Endgerätes. Verschiedene Ortungsdienste wie GPS, WLAN-Ortung, Mobilfunkortung mit ihren verschiedenen Eigenschaften und Möglichkeiten werden abstrahiert durch sogenannte *LocationProvider*. Je nachdem was das Gerät unterstützt, stehen keiner oder mehrere dieser LocationProvider zur Verfügung.

Die Auswahl eines LocationProviders erfolgt entweder manuell, wobei es in der Verantwortung des LBS liegt, selbst auszuwählen oder die Entscheidung dem Endnutzer zu überlassen, oder automatisch durch den LocationManager. Dazu kann man Kriterien definieren wie minimale Genauigkeit (in Meter), Höhenangabe erforderlich, Kosten (für Endnutzer) erlaubt.

Es gibt die Möglichkeit, die letzte bekannte Position eines jeweiligen LocationProviders abzufragen. Das hat den Vorteil, dass keine (Geld- und Energie-) Kosten entstehen, da der Service nicht

[38] http://www.android.com/ (Stand: 29.06.2010)
[39] http://www.openhandsetalliance.com/ (Stand: 29.06.2010)

aktiviert werden muss. Die Information kann allerdings veraltet oder (noch) garnicht vorhanden sein.

Die Standortdaten enthalten die Koordinaten (Latitude, Longitude) und falls vom LocationProvider unterstützt Genauigkeit (Accuracy), Höhe (Altitude), Ausrichtung (Bearing) und Geschwindigkeit (Speed).

Außerdem ist es möglich Push-Services zu realisieren. Unter Angabe von Koordinaten und Umkreisradius kann man Benachrichtigungen registrieren. Sobald man einen so spezifizierten Bereich betritt wird ein Ereignis ausgelöst. Intern werden dabei allerdings GPS- und Netzwerkprovider zur Positionserkennung benutzt, ohne darauf weiteren Einfluss nehmen zu können.

Bewertung Die Location API von Android bietet einen einheitlichen Ansatz, auf die Standortinformationen mobiler Android-Endgeräte zuzugreifen. Durch die Spezifikation von Kriterien wie Genauigkeit oder Kosten wird die automatisierte Auswahl von Ortungsdiensten ermöglicht.

Die in dieser Arbeit gestellten Anforderungen an eine Ortungsmiddleware werden aber nicht oder nur ansatzweise erfüllt. So ist es nicht möglich neue LocationProvider zu erstellen und dynamisch neue Ortungsdienste hinzuzufügen. Wie die Auswahl auf Basis der Kriterien erfolgt ist unbekannt und erfolgt wie bei JSR-179 nur zum Instanziierungszeitpunkt. Auswahl mehrerer Ortungsdienste und Fusion der Daten ist nicht möglich. Symbolische Standortinformationen werden ebenfalls nicht unterstützt.

4.4.3 Forschungsansätze - Positioning Integration Middlewares

4.4.3.1 PoSIM: Ansatz von Bellavista, Corradi, Giannelli

Die Positioning Integration an Management Middleware (PoSIM) [BCG08] verfolgt das Ziel der synergetischen Nutzung und Kontrolle heterogener Lokalisierungssysteme zur Förderung der Portabilität von LBS-Anwendungen. Dabei wird ein anderer Ansatz als typische Middlewarelösungen verfolgt, welche die zugrunde liegenden Mechanismen und Charakteristiken verstecken, die aber wie ausgeführt wird, wichtig sind für “smarte” LBS.

So gibt es einen High-Level Ansatz für den typischen transparenten Zugriff auf Lokalisierung sowie einen Low-Level-Ansatz, der es LBS erlaubt, die zu Grunde liegenden Lokalisierungssysteme direkt zu steuern. Beim High-Level-Ansatz erfolgt die dynamische Kontrolle der Lokalisierungssysteme durch Richtlinien, Events und Filter. Durch geschickte Kombinierung lässt sich zum Beispiel erreichen, dass alle Systeme mit hohem Energiebedarf abgeschaltet werden aber trotzdem sichergestellt ist, dass das System mit der höchsten Genauigkeit aktiv bleibt.

Die Steuerung, welche Systeme unter welchen Bedingungen ausgeführt werden, geschieht durch Richtlinien (Policies). Dies wirkt sich auf alle darüber liegenden LBS aus. Erstellung und Wartung

der Richtlinien ist ein Administrativer Vorgang. Aktivierung hingegen erfolgt durch die LBS. Allerdings gibt es keine Kontrolle und Konfliktlösung. LBS A könnte eine spezielle Richtline aktivieren und LBS B könnte sie wieder deaktivieren. Ein sammeln von LBS-Anforderungen, und dynamische Einstellung der Richtlinien ist nicht vorgesehen. Durch Filter, die jede LBS selbst definiert ohne die anderen zu beeinflussen, wird nur die Rückgabe der zentral ermittelten Standortdaten angepasst, um Overhead zu reduzieren und unwichtige Standortupdates zu vermeiden.

Die Lösung mangelt an dynamischer Entdeckung neuer Lokalisierungssysteme. Zwar lassen sich Systeme, für welche ein Wrapper implementiert wurde an PoSIM anmelden, ein dynamisches Registrierung mit Dienstbeschreibung und Entdeckung scheint es nicht zu geben. Außerdem ist es zwar möglich die Standortdaten mehrerer Systeme zu sammeln und über eine einheitliche Ontologie weiterzuleiten. Eine Datenfusion der ermittelten Standortdaten zur Erhöhung der Genauigkeit ist nicht vorgesehen. Stattdessen werden die Standortdaten der verschiedenen genutzten Ortungsdienste in einer XML-Datei zusammengefasst.

Der Kern des Systems, welcher die verschiedenen Richtlinien zur Aktivierung und Deaktivierung von Ortungsdiensten verarbeitet lässt sich zwar dynamisch konfigurieren, Auswahl und Aktivierung liegt allerdings in der Hand der LBS. Dies kann zu Konflikten führen. Eine Aggregation der Anforderungen verschiedener LBS und Einstellung der Richtlinien auf dieser Basis ist nicht möglich.

4.4.3.2 Filjar, Busic, Desic, Huljenic

In [FBDH08] wird ein System vorgeschlagen, welches adaptiv, auf Basis angeforderter QoS-Eigenschaften Lokalisierungsmethoden auswählt. Kernpunkt ist ein iterativer Algorithmus. Über ein Nutzerprofil wird die Liste der unterstützten und vom Nutzer bevorzugte oder erlaubten Methoden bereitgestellt. Über ein Serviceprofil werden die minimalen QoS-Eigenschaften des jeweiligen LBS spezifiziert, konkret die zu akzeptierende horizontale und vertikale Genauigkeit und Antwortzeit.

Der iterative Algorithmus prüft die QoS-Anforderungen gegen gelieferte Ergebnisse der konkrete Methoden. Zunächst werden die aktiv laufenden Methoden geprüft, in weiteren Schritten werden zusätzliche Methoden aktiviert, bis die Anforderungen erfüllt werden, oder keine Verbesserung mehr erreicht werden kann. Ermittelte Position (Länge, Breite, optional Höhe) werden an den LBS übermittelt. Die ermittelten QoS-Eigenschaften (horizontaler/vertikaler Fehler und Antwortzeit) werden nur zur internen Verarbeitung genutzt.

Der iterative Algorithmus stellt einen vielversprechenden Ansatz dar und ist notwendig, da a priori die lieferbaren Ergebnisse einer spezifischen Methode oft nicht ermittelbar sind. Leider wird von diesem System das dynamische Finden neuer, unbekannter Methoden nicht unterstützt.

Datenfusion wird zwar erwähnt und ist offenbar theoretisch möglich, nähere Details gibt es aber nicht. Der Zusammenhang der QoS-Spezifikation im Serviceprofil und möglichen Anforderungen eines LBS sind unklar. Neben der ermittelten Position sollten die Qualitätseigenschaften außerdem ebenfalls an den LBS zurückgeliefert werden.

4.4.3.3 Adaptive LBS: Ansatz von Horbank und Ibach

Horbank und Ilbach [Hor07, IH05] verfolgen einen interessanten Ansatz zur Realisierung hochgradig adaptiver LBS. Dabei wird nicht nur die Lokalisierung betrachtet, sondern die gesamte LBS-Beschaffungskette. Jede funktionale Komponente (zum Beispiel mobile Verbindung, Lokalisierung, semantische Standortbestimmung, Inhalt und Karten, Abrechnung) wird durch eine Dienstklasse repräsentiert. Zur Laufzeit des LBS können dann über Dienstverzeichnisse passende Dienstinstanzen gesucht, gefunden und schließlich eingebunden werden. Der LBS wird also adaptiv und dynamisch zusammengesetzt.

Dazu wurde auf die Möglichkeiten von Web Services zurückgegriffen. Jede LBS-Komponente wird durch eine Web-Service-Klasse (Porttyp) repräsentiert. Zum Auffinden der Dienste gibt es einen speziell angepassten Verzeichnisdienst, den Location-based Discovery Service (LBDS).

Interessant ist die Nutzung von WS-Discovery, womit sich verfügbare Dienste in lokalen Netzwerken veröffentlichen lassen. So können beispielsweise bei Einwahl in das WLAN eines Gebäudes, dort verfügbare Dienste erkannt werden.

Zur Auswahl eines Dienstes aus der Klasse zur Lokalisierung werden verschiedene gewichtete Kriterien wie Reichweite, Latenz, Qualität, Kosten und Nutzerfeedback herangezogen.

Der adaptive Ansatz, LBS-Komponenten als austauschbare Dienste zu interpretieren und dynamisch zu suchen, finden und binden ist sehr gut. Die Implementation durch Web Services garantiert Interoperabilität und Flexibilität. Für den Einsatz ressourcenschwachen Smartphones ist sie allerdings kritisch zu sehen.

Darüber hinaus sollte noch einen Schritt weitergegangen werden. Der Auswahlalgorithmus sollte wesentlich flexibler sein bezüglich der LBS-Anforderungen. Auswahl mehrerer Ortungsdienste, Möglichkeit zur Datenfusion und Einbindung von Bewegungssensoren könnten die Qualität verbessern.

4.4.3.4 Indoor-Outdoor-Handover von Hansen, Wind, Jensen, Thomsen

In [HWJT09] wird die Implementation einer Ortungslösung vorgestellt, welche den nahtlosen Übergang von Outdoor zu Indoor ermöglicht. Dabei wird transparent umgeschaltet zwischen GPS-Ortung im Außenbereich und WLAN-Ortung im Innenbereich. Vier verschiedene Strategien werden

vorgestellt und ausgewertet, welche sich darin unterscheiden, welche Methode unter welchen Bedingungen präferiert wird, und wann eine Umschaltung ausgelöst wird. Unter besonderer Beobachtung stehen dabei die Genauigkeit der Ortung, sowie der Energieverbrauch. Bestes Ergebnis erzielte die Strategie, welche primär GPS benutzt solange fortwährend fünf Sekunden lang eine gültige Position ermittelt wurde. Erst wenn dies nicht mehr gegben ist, gleichbedeutend damit, dass GPS unzuverlässig geworden ist, wird auf WLAN umschaltet.

4.4.3.5 MiddleWhere

MiddleWhere [RAC+04]ist keine Ortungsmiddleware für mobile Endgeräte und mobile LBS im eigentlichen Sinn, sondern vielmehr eine Location-aware Kontextsystem. Es integriert viele verschiedene, verteilte Ortungssysteme und Sensoren und bietet Anwendungen eine zusammengefasste Sicht auf den Standort mobiler Objekte (Personen oder Geräte). Standortdaten können bezogen werden aus RF-basierten Ausweisen, Magnetstreifenkarten, Login-Informationen von Benutzeroberflächen, Fingerabdruckscannern, Bluetooth, GPS.

Das System verarbeitet Koordinaten und symbolische Standortdaten, unterstützt Push und Pull Interaktion. Es enthält außerdem ein Modell der Welt, um die räumlichen Beziehungen der Umgebung wie Gebäude, Räume, Bereiche, Personen und Objekte zu erfassen.

Das System ist mit einer Location Middleware wie sie in dieser Arbeit behandelt wird nicht direkt zu vergleichen. Dennoch ist das Prinzip der Behandlung der Standortdaten und Fusion dieser Daten sehr interessant. Die Middleware misst die Qualität der Standortdaten über die Auflösung (Genauigkeit), Zuverlässigkeit und Aktualität. Mit der Aktualität lässt sich die temporale Eigenschaft von Ortsinformationen behandeln, denn je länger eine Messung zurückliegt, umso unwahrscheinlicher ist, dass sich die Person noch dort aufhält. Schließlich wird aus den Daten der verschiedenen Sensoren die räumliche Wahrscheinlichkeitsverteilung der Position einer Person berechnet.

4.5 Zusammenfassung

LBS und Standortbestimmung sind hochaktuelle Themen, die in den letzten Jahren sowohl im kommerziellen als auch wissenschaftlichen Bereich ausgiebig untersucht wurden. Der Trend zu Flexibilität und Dynamik zeichnet sich ab. Viele kommerzielle Anbieter integrieren verschiedenste Ortungsdienste in eine gemeinsame Lösung. Mit der Geolocation-API wird es erstmals die Brücke von Webapplikationen zu gerätebasieren Ortungsmethoden geschlagen, in dem eine standardisierte Schnittstelle geschaffen wurde.

Aus in der Forschung wurden verschiedene Ansätze und Aspekte untersucht, Ortungsdienste dynamisch zu finden und auszuwählen, oder Standortdaten von verschiedenen Quellen zu ver-

schmelzen. Einen einheitlichen Ansatz einer Ortungsmiddleware wie sie am Anfang dieses Kapitel skizziert wurde gibt es jedoch scheinbar noch nicht.

Bei einigen Aspekten ist zudem aufgefallen, dass Entwürfe zwar getestet wurden, allerdings auf "gewöhnlichen" Notebooks, und nicht auf den ressourcenbegrenzten Smartphones. Praxistauglichkeit ist damit fraglich.

5 Konzept

Im vorangegangenen Kapitel wurde die Problemstellung dieser Arbeit genauer analysiert. Es wurden Beispielszenarien vorgestellt und die Probleme erläutert die sich bei dynamischer Suche, Auswahl und Kombination verschiedenartiger Ortungsdienste ergeben. Anschließend wurden Anforderungen abgeleitet welche von einer Ortungsmiddleware erfüllt werden müssen. Mit diesem Fokus wurden verschiedene Ansätze, sowohl kommerzieller als auch standardisierter Art und aus der Forschung vorgestellt.

Im folgenden Kapitel gilt es, die gesammelten Erkenntnisse zu nutzen um ein Konzept für eine Ortungsmiddleware zu erstellen. Entsprechend den Anforderungen aus Kapitel 4.3 soll eine Architektur entwickelt werden, welche die Erfüllung der genannten Aufgaben ermöglicht.

Besonderes Augenmerk soll auf die Erweiterungsmöglichkeit gelegt werden, da für jeden Aspekt verschiedene Umsetzungen und Strategien denkbar sind. Daher empfiehlt es sich, das System modular aufzubauen. Die jeweiligen Funktionseinheiten können dann zu einem späteren Zeitpunkt beliebig angepasst, erweitert oder ausgetauscht werden.

5.1 Architektur

Die Ausführung von LBS-Anwendungen unter Nutzung der Ortungsmiddleware findet auf drei Ebenen statt. Oben befinden sich LBS, welche Anfragen mit unterschiedlichen Anforderungen an die Standortbestimmung und Standortinformationen stellen.
Unten befinden sich die verschiedenen Ortungsdienste, welche Unterschiede in Verfügbarkeit und Eigenschaften der lieferbaren Standortinformationen aufweisen.
Dazwischen befindet sich die Ortungsmiddleware. Sie muss die Anfragen entgegennehmen und auswerten, die verfügbaren Ortungsdienste erkennen und auswerten, und schließlich Angebot und Nachfrage in Beziehung setzen um eine Auswahl und Aktivierung bestimmter Ortungsdienste durchzuführen, Standortinformationen zu sammeln, zu kombinieren und an die LBS zurückzugeben.

Die folgenden Komponenten bilden das Grundgerüst der Ortungsmiddleware: eine Schnittstelle (API) für LBS, verschiedene Ortungsdienste (LocationService LCS), ein Manager zum verwalten der Ortungsdienste (LocationServiceManager LCSM), ein Manager zum Verwalten der ermittelten Standortinformationen (DataManager), die zentrale Kernkomponente mit verschiedenen

Strategien für Dienstauswahl und Standortdatenverarbeitung (LocationEstimator LE) und einer Nutzerschnittstelle zur Konfiguration des Verhaltens der Middleware. In Abbildung 5.1 wird die Architektur in einer Übersicht dargestellt.

Die Middleware soll dabei keine Möglichkeit haben, Systemfunktionen zu aktivieren oder deaktivieren. Die Kontrolle liegt vollständig in der Hand des Nutzers (Gerätebesitzer). Er steuert über Profil- und Systemeinstellungen die Möglichkeiten der Middleware und der zu Grunde liegenden Ortungsdienste.

5.1.1 Schnittstelle zu LBS

Um LBS mit Standortinformationen versorgen zu können muss eine Schnittstelle (API) bereitgestellt werden. Über diese stellen LBS Anfragen mit spezifizierten Anforderungen und erhalten Anworten mit den ermittelten Standortdaten.

5.1.1.1 Anfragen

Eine LBS-Standortanfrage enthält unterschiedliche Daten, welche sich auf verschiedene Aspekte beziehen.

Funktionale Anforderungen Der wichtigste Teil einer Standortanfrage (*LocationRequest*) sind die funktionalen Anforderungen. Sie bestehen aus dem Typ der Anfrage und dem gewünschten Format.

Anfragetyp (*RequestType*) spezifiziert die Art der Anfrage: einfach (*single*), periodisch (*periodic*) oder proaktiv (*push*). Bei einer einfachen Anfrage wird lediglich eine einmalige Antwort mit Standortinformationen benötigt. Nach erfolgreicher Übermittlung der Standortdaten ist der Vorgang beendet.
Bei periodischen Standortupdates muss vom LBS ein Updateintervall vorgegeben werden. Die Standortinformationen werden dann kontinuierlich ermittelt und im vorgegebenen Intervall an den LBS zurückgegeben. Der Vorgang endet erst, wenn die Anfrage vom LBS aufgehoben wird.
Die dritte Möglichkeit besteht darin, Registrierungen für proaktive Standortupdates zu erstellen, zum Beispiel durch Angabe von verschiedenen Bereichen. Die Middleware ermittelt daraufhin kontinuierlich den Standort. Sobald ein registrierter Bereich betreten wird, wird der LBS benachrichtigt. Der Vorgang endet ebenfalls erst wenn der LBS die Anfrage aufhebt.

Standortformat (*DataType*) spezifiziert das Standortformat, welches vom LBS benötigt wird. Möglich sind Koordinaten (*coordinates*), bestehend aus Länge, Breite (*latitude*, *longitude*)

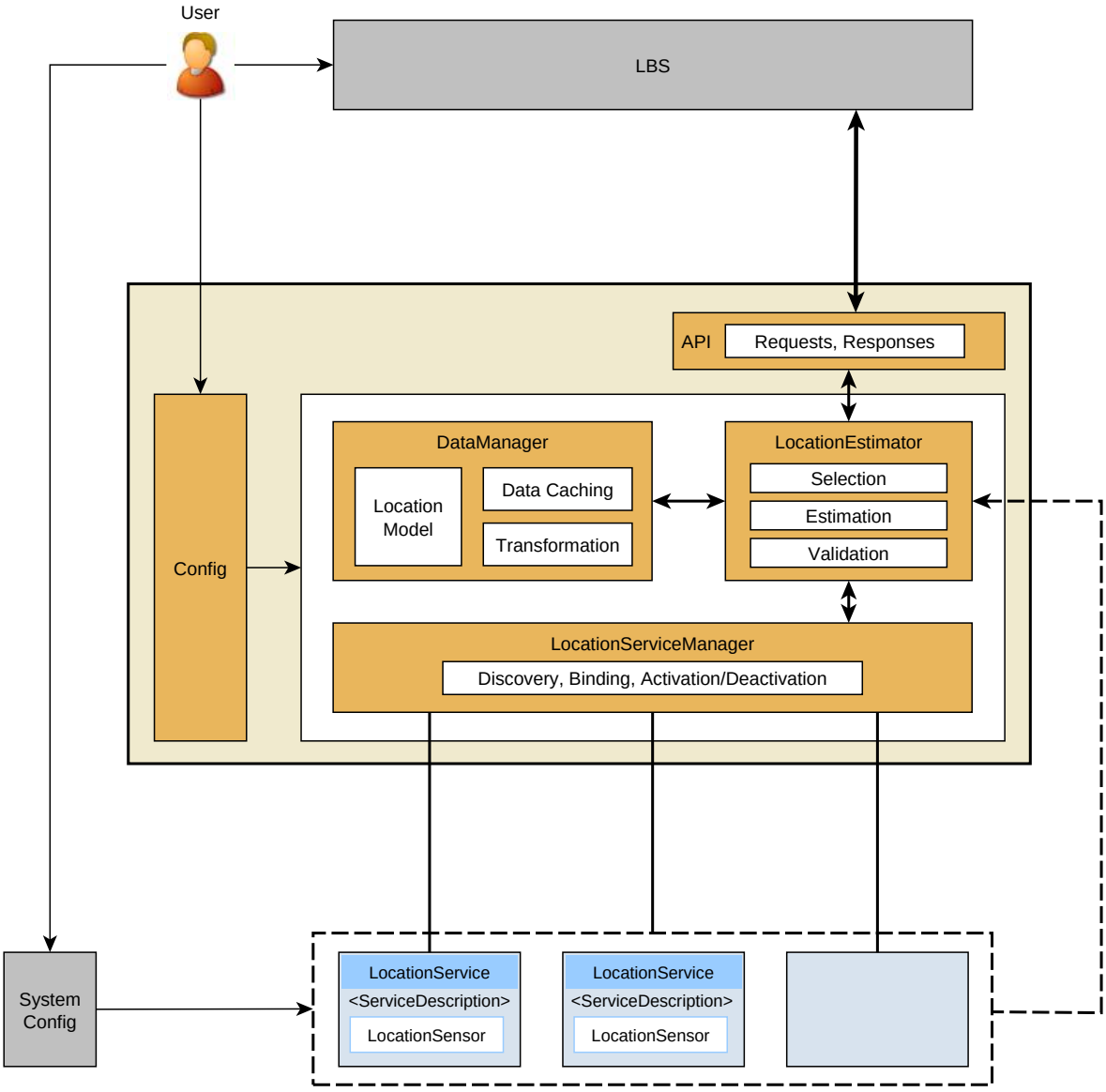


Abb. 5.1: Grundlegende Architektur Ortungsmiddleware

und optional Höhe (*altitude*), symbolische Standortdaten (*symbolic*) oder Bewegungsdaten (*movement*) wie Geschwindigkeit (*speed*) und Richtung (*direction*).

Die funktionalen Anforderungen sind sogenannte *hard requirements*, d.h. sie müssen erfüllt werden, soweit es auf Grund der verfügbaren Ortungsdienste möglich ist. Damit lässt sich die Menge der verfügbaren Ortungsdienste für die spätere Auswahl und Aktivierung bereits eingrenzen. Dienste, welche geforderte Typen oder Formate nicht unterstützen, oder deren Daten durch weitere middlewareinterne Verarbeitung nicht an die Anforderungen angepasst werden können, werden für die weiteren Auswahlschritte nicht weiter betrachtet.

Middlewareinterne Verarbeitung bedeutet, dass ein gewünschtes Feature (Typ oder Format) nicht vom Ortungsdienst selbst, sondern von einer Komponente der Middleware realisiert wird. Stehen beispielsweise keine Ortungsdienste zur Verfügung, welche proaktive Anfragen unterstützen, könnte die Middleware selbst Registrierungen speichern und periodisch Standortdaten ermitteln und abgleichen. Stehen keine Ortungsdienste mit den geforderten Standortformaten zur Verfügung besteht evtl. die Möglichkeit durch Transformation das gewünschte Ergebnis zu erzielen.

Nichtunktionale (qualitative) Anforderungen Der zweite Teil einer Standortanfrage bezieht sich auf die qualitativen Anforderungen an die Standortermittlung und die Standortinformationen. Dabei ist wiederum zu unterscheiden in Anforderungen an das Ergebnis, also die Standortinformationen selbst (Genauigkeit, Aktualität, Zuverlässigkeit), sowie an den Vorgang der Standortbestimmung (Kosten, Aufwand, Verzögerungen). Die qualitativen Anforderungen kann man auch als *soft requirements* bezeichnen. Auch wenn sie nicht vollständig erfüllt werden können, so wird doch die bestmögliche Zielerreichung angestrebt.

Genauigkeit (*Accuracy*) der gewünschten Standortinformationen. Sie sollte sich auf das geforderte Standortformat beziehen, typischerweise auf die geografische Position.

Aktualität (*Freshness*) der Standortdaten gibt Auskunft darüber, wie lang die Ermittlung der Daten schon zurückliegt.

Nicht bei jedem Anfragetyp muss auf Grund des unterschiedlichen Verhaltens jeder Qualitätsparamter vom LBS spezifiziert werden. Es sollte außerdem möglich sein, auf die Angabe einiger Parameter verzichten zu können, zum Beispiel wenn zwar eine bestimmte Genauigkeit wichtig ist, eine Verzögerung aber unwichtig erscheint.

Für einfache Anfragen sind beide Parameter zu spezifizieren. Es ist wichtig, welche Genauigkeit gefordert wird und welches maximale Alter die Standortinformationen haben dürfen.

Bei periodischen Anfragen muss beachtet werden, dass Aktualität und Intervall zusammenpassen. Das maximal zulässige Alter der Standortinformationen darf logischerweise nicht größer sein, als das Intervall.

Bei proaktiven Standortupdates erfolgt die Angabe der Genauigkeit in Kombination mit den Registrierungsdefinitionen und sagt damit aus, wie sensibel die Ereignisauslösung erfolgt. Da proaktive Updates Echtzeitcharakter haben, wird implizit eine hochgradige Aktualität gefordert.

5.1.1.2 Antworten

Die Antworten (*LocationResponse*) bestehen aus den ermittelten Standortdaten, versehen mit den ermittelten Qualitätsparametern und Statusinformationen.

Da es möglich sein soll, dass verschiedene LBS, unterschiedliche Formate anfordern können, sollten die jeweiligen Anworten auch nur Standortinformationen in diesen Formaten enthalten, um Overhead zu vermeiden.

Passend zu den Formaten gehören die entsprechenden Qualitätsinformationen wie Genauigkeit, Aktualität und Zuverlässigkeit.

Um den qualitätsbewussten Ansatz zu ermöglichen werden außerdem Statusupdates übermittelt, welche Auskunft darüber geben, ob ein Bestimmungsvorgang bereits abgeschlossen ist oder nicht. Das kann sinnvoll sein, wenn nicht alle Anforderungen umgehend und im ersten Auwahlversuch erfüllt werden können. Eine Anfrage mit hoher Genauigkeit und niedriger Verzögerung kann so zum Beispiel zunächst umgehend ein Standortupdate mit niedriger Genauigkeit erhalten, während versucht wird, eine länger dauernde Bestimmung mit hoher Genauigkeit durchzuführen.
Der LBS kann dann selbst entscheiden, ob er diese Zwischeninformation verarbeitet oder auf das finale Standortupdate wartet. Er kann den Versuch auch vorzeitig abbrechen.

5.1.2 LocationService (LCS)

Jede Ortungsmethode oder Sensor für zusätzliche Standortinformationen (*LocationSensor* LS) wird durch einen Ortungsdienst (*LocationService* LCS) repräsentiert. Der LCS kapselt (eng.: wrapped) dabei die systemspezifischen Eigenschaften der zu Grunde liegenden Ortungslösung und stellt eine einheitliche Schnittstelle für die Middleware zur Verfügung. Damit lassen sich alle Arten von Ortungslösungen einbinden, die Standortinformationen bereitstellen wie zum Beispiel lokale Lösungen auf Basis GPS, WLAN, inertialer Sensoren oder netzwerkbasierte Ortungsdienste.

5.1.2.1 Fähigkeiten und Dienstbeschreibung

Jeder LCS und die damit gekapselte Ortungslösung hat verschiedene Fähigkeiten und Eigenschaften, welche sich auf unterschiedliche Aspekte beziehen.

Wie bereits in Kapitel 4.2.1 beschrieben wurde, gibt es eine Reihe von Qualitätseigenschaften, welche hochgradig dynamisch sind und von der momentanen Umgebung (Situation, Position)

des Nutzers abhängen. Solche Eigenschaften lassen sich nicht im Voraus statisch beschreiben und eindeutig festlegen. Es lassen sich nur, auf Grund von Beobachtungen ermittelte, typische Richtwerte, oder Werte für den Optimalfall spezifizieren. Welche Werte schließlich tatsächlich zutreffen, lässt sich erst durch tatsächliche Aktivierung und Ermittlung feststellen.

Das Erfassen und Verarbeiten dieser Dynamik, das bestmögliche Abschätzen im Voraus, Auswahl, Ermittlung und Prüfung stellen die besondere Herausforderung der Ortungsmiddleware dar.

Statische Eigenschaften Statische Eigenschaften eines Ortungsdienstes lassen sich eindeutig und im Voraus festlegen und ändern sich gewöhnlich nicht.

Dazu zählen zunächst die *funktionalen* Eigenschaften wie unterstützte **Standortformate** und **Anfragetypen** (Siehe Kapitel 5.1.1.1)

Zu den *nichtfunktionalen* (qualitativen) statischen Eigenschaften zählen:

Latenz (*Latency*) ist die Zeit von der Anfrage an einen Ortungsdienst, bis die Standortinformationen bereit stehen. Daraus ergibt sich das kleinstmögliche Abfrageintervall für kontinuierliche Standortupdates.

Kosten (*Charges*) gibt Auskunft darüber, ob die Nutzung eines Ortungsdienste (monetäre) Kosten verursacht. Es kann eine Unterscheidung in ja oder nein getroffen oder eine genaue Angabe über die Höhe festgelegt werden.

Aufwand (*Costs*) ist eine zusammengefasste Bewertung des Aufwandes (= Kosten) welcher durch Overhead und Energieverbrauch entsteht.

Dynamische Eigenschaften Dynamische Eigenschaften lassen sich nicht eindeutig und im Voraus festlegen, da sie zur Laufzeit, je nach Situation und Position des Nutzers variieren können. Erfasst wird zunächst nur ein a priori-Wert, welcher den typischen, optimalen Fall kennzeichnet. Er kann für einen ersten Auswahlschritt benutzt werden, muss aber im weiteren Verlauf mit tatsächlich ermittelten Daten korrigiert werden.

Genauigkeit (*Accuracy*) der lieferbaren Standortinformationen.

TTFF (Time to first fix) spezifiziert die Zeit von der Anfrage bis die erste Standortinformation bereit steht.

Vor allem bei Ortungsdiensten mit niedriger Konsistenz hängt die Genauigkeit stark von der aktuellen Position ab. GPS ereicht unter günstigen Bedingungen eine Genauigkeit von bis zu 5 Metern. Sobald aber die Sichtverbindung zu ausreichend Satelliten eingeschränkt ist, nimmt die Genauigkeit stark ab. Bei Mobilfunkortung richtet sich die Genauigkeit nach der Zellengröße und schwankt von einigen hundert Metern im städtischen Bereich bis zu mehreren Kilometern auf dem Land.

Die TTFF hängt von verschiedenen Faktoren ab. Bei GPS zum Beispiel, wie lang die letzte Standortbestimmung zurücklag, und ob Almanachdaten (sind in den Signalen der Satelliten enthalten) erst aktualisiert werden müssen oder nicht. Ob eine Internetverbindung besteht und dadurch möglicherweise A-GPS nutzbar ist wirkt sich ebenfalls aus.

Weitere Eigenschaften

Lokal oder global LCS lassen sich klassifizieren als globale oder lokale Dienste. Globale Dienste wie GPS oder Mobilfunkortung sind prinzipiell überall verfügbar. Lokale Dienste sind nur innerhalb eines bestimmten Gebietes verfügbar, zum Beispiel eine WLAN-Ortungssystems auf einem Campus oder in einer Einkaufspassage. Es ist zu vermuten, da lokale Lösungen an die jeweilige Umgebung besonders gut angepasst sind, diese auch ein höheres Qualitätspotential haben als globale Dienste. Lokale Dienste könnten daher höher priorisiert werden als lokale.

Allgemeine Zuverlässigkeit Aus vielen Experimenten und grundsätzlichen Eigenschaften könnte man einen allgemeinen Zuverlässigkeitswert für einen Dienst ableiten.

Dienstbeschreibung Alle denkbaren Eigenschaften und Fähigkeiten eines Ortungsdienstes müssen in einer Dienstbeschreibung zusammengefasst und verfügbar gemacht werden, zum Beispiel in Form eines XML-Dokuments. Bei allen Verarbeitungsschritten innerhalb der Middleware wie Auswahl, Qualitätsbewertung und Verarbeitung der Locationupdates können und sollten diese Informationen dann ausgewertet werden. Tabelle A.2 im Anhang fasst mögliche Parameter noch einmal in einer Übersicht zusammen.

5.1.2.2 Letzter Standort

Der LCS sollte die zuletzt ermittelten Standortdaten zwischenspeichern. So kann bei einer späteren neuen Anfrage zunächst der letzte Standort ausgelesen werden, ohne den Dienst zu aktivieren und auszuführen. Der Wrapper muss diese Informationen entweder selbst speichern, oder wenn möglich direkt vom Sensor abfragen.

5.1.2.3 Zustand

Jeder Ortungsdienst hat verschiedene Zustände, welche Informationen seine Nutzbarkeit geben. Zwei Eigenschaften wirken sich auf den Zustand aus: Bedingungen (requirements) und ob er (von der Middleware) ausgewählt und damit gestartet oder gestoppt wurde. Abbildung 5.2 zeigt die Zustände und Übergänge.

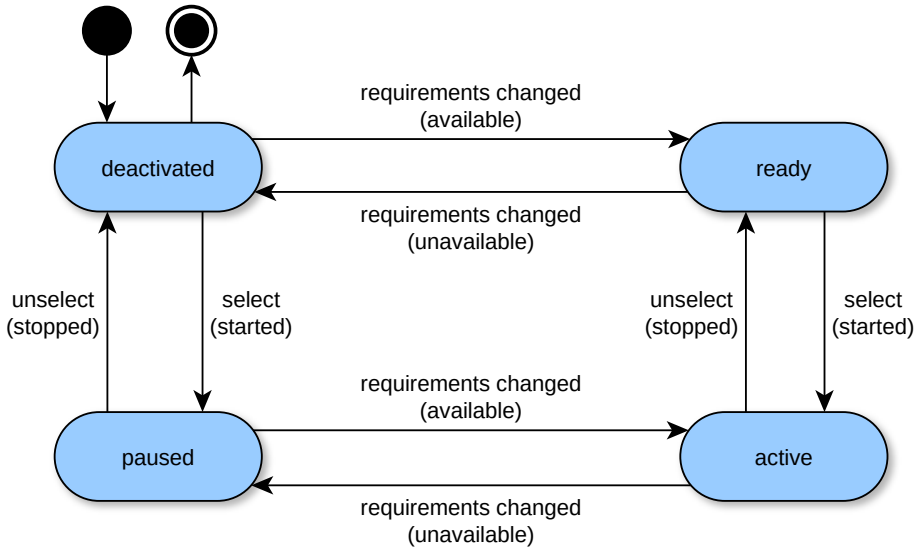


Abb. 5.2: Zustandsübergänge der Location Services

Deaktiviert (*deactivated*) Der Dienst ist momentan nicht nutzbar, da Voraussetzungen nicht erfüllt sind. Er ist damit deaktiviert. Grund dafür sind technische Voraussetzungen wie Gerätefunktionen oder Restriktionen durch den Anwender, z.B. GPS-Modul deaktiviert, Mobilfunkverbindung getrennt, WLAN deaktiviert, Verwendung durch Nutzer gesperrt.

Bereit (*ready*) Der Dienst ist momentan nicht aktiv, aber alle Voraussetzungen sind erfüllt und er kann jederzeit ausgewählt und gestartet werden.

Aktiv (*active*) Der Dienst wurde ausgewählt, ist aktiv und Standortdaten werden ermittelt bzw. übertragen.

Pausiert (*paused*) Obwohl ein Dienst ausgewählt und aktiviert wurde, ist eine Voraussetzung nicht mehr gegeben, der Dienst ist daher pausiert. Er kann gestoppt werden, oder es wird gewartet, bis wieder alle Voraussetzungen erfüllt sind.

Jeder LCS ist selbst dafür verantwortlich, die Systemzustände zu erkennen, die seine Ausführung voraussetzen, und entsprechend seinen Zustand aktuell zu halten. Zustandsänderungen werden wiederum an die Middleware weitergegeben, damit diese darauf mit einer Aktualisierung der Dienstauswahl reagieren kann.

Neben der Information über den Status und seine Änderung ist es außerdem sinnvoll, auch die Information über die Voraussetzungen weiterzuleiten, damit diese bei Bedarf an den LBS

oder Endnutzer weitergereicht werden können und dieser aktiv werden kann. Ist zum Beispiel das GPS-Modul deaktiviert, könnte diese Information an den Endnutzer weitergereicht werden, welcher dann selbst entscheiden kann, ob er es reaktiviert oder nicht.

5.1.2.4 Lokale und entfernte Ortungsdienste

In Kapitel 3.2.1 wurde auf die Eigenschaft eines Ortungsdienstes eingegangen, wo die Bestimmung des Standortes erfolgt und der Zugriff ermöglicht wird. Dabei wurden netzwerkbasierte, gerätebasierte und hybride Ortungsdienste unterschieden. Dementsprechend kann der Zugriff über eine Schnittstelle auf dem Endgerät oder im Netzwerk erfolgen. Für die Middleware soll das grundsätzlich keine Rolle spielen. Das Zusammenführen von entfernten und lokalen Diensten und die Auseinandersetzung mit den Spezifika der verschiedenen Anbindungsmöglichkeiten erfolgt mit Hilfe des *LocationServiceManagers* bzw versteckt durch die LocationService-Wrapper.

5.1.3 LocationServiceManager (LCSM)

Der *LocationServiceManager* (LCSM) bildet das Bindeglied zwischen der Middleware und den verschiedenen abstrahierten Ortungsdiensten. Er ist für das Auffinden und Anbinden verfügbarer Ortungsdienste verantwortlich und erfüllt daher die Rolle eines Dienstverzeichnis. Er muss außerdem Statusänderungen der Ortungsdienste überwachen und an die Middleware weiterleiten.

Er liefert der Middleware auf Anfrage Informationen über die zurzeit auf dem Endgerät verfügbaren Ortungsdienste. Dazu muss er von jedem bekannten Ortungdsdienst die Dienstbeschreibung vorhalten. Auf dieser Basis kann er Anfragen der Middleware filtern, zum Beispiel nach Diensten mit bestimmte Format.

Der LocationServiceManager stellt die Schnittstellendefinition bereit, welche von den Ortungdsdiensten (LCS) implementiert werden muss. Damit ist eine einheitliche Anbindung verschiedener Ortungsdienste gesichert und das System um neue, bisher unbekannte Ortungdsdienste erweiterbar. Um eine neue Ortungslösung verwenden zu können, muss nur ein neuer Wrapper entwickelt werden, der die vom LCSM vorgegebene Schnittstelle implementiert, und auf dem Endgerät installiert werden.

5.1.4 LocationEstimator (LE)

Der *LocationEstimator* (LE) ist die zentrale Hauptkomponente der Middleware und das Bindeglied zwischen Standortanfragen der LBS und Standortangebot der Ortungsdienste. Er enthält die Geschäftslogik der Middleware und steuert den Prozess von der Anforderungsauswertung, über Ortungsdienstauswahl, Bewertung ermittelter Standortinformationen, Datenaggregation und Standortrückgabe. In Kapitel 5.3 werden die Vorgänge genauer erläutert.

Folgende Aufgaben werden erfüllt:

- Hinzufügen/Entfernen von LocationRequests
- Beantworten von LocationRequests (Übertragung von Standortinformationen und Statusinformationen)
- Informationen über verfügbare LCS und ihre Eigenschaften vom LCSM abfragen und auswerten
- LCS auswählen oder abwählen.
- Von LCS erhaltene Standortdaten aggregieren

5.1.5 LocationDataManager (LDM)

Der *LocationDataManager* (DM) ist für die Verarbeitung und Speicherung der Standortinformationen verantwortlich. Er speichert für jeden genutzten Ortungsdienst eine Reihe der letzten ermittelten Standortinformationen zwischen. Außerdem speichert er den durch Aggregation ermittelten zentralen Standortdatensatz (*Basiclocation*).

Er stellt Methoden zur Verfügung um verschiedene Standortformate zu transformieren, zum Beispiel um aus einem symbolischen Standortdatum die zugehörigen Koordinaten zu ermitteln.

Folgende Aufgaben werden erfüllt:

- Cached ermittelte Standortdaten jedes einzelnen LCS
- Speichert zentralen Standortdatensatz (Basiclocation)
- Stellt LocationModel zur Verfügung
- bietet Methoden zur Transformation

5.1.6 UserInterface (UI)

Damit der LBS-Nutzer (Gerätebesitzer) die Möglichkeit hat, das Verhalten der Middleware zu steuern, greifen an verschiedenen Punkten grafische Nutzerschnittstellen (*UserInterface* UI) ein.

Ein Punkt betrifft das Einsehen und Verwalten der vom LCSM gefundenen Ortungsdienste. Es soll möglich sein, einzelne Dienste zu erlauben oder verbieten, oder eine eigene Priorisierung vornehmen zu können.

Ein zweiter Punkt betrifft die Zugriffskontrolle für LBS. Der Nutzer sollte anfragenden LBS den Zugriff auf von der Middleware ermittelte Standortinformationen erlauben oder verbieten können. Denkbar ist auch, unterschiedliche Qualitätstufen für verschiedene LBS vorgeben zu können.

Der letzte Punkt betrifft das grundsätzliche Verhalten der Kernkomponente, das heißt der Nutzer könnte zwischen den verschiedenen Auswahl- und Aggregationsstrategien auswählen.

5.1.7 RemoteLocationServiceManager (RLCSM)

Der *RemoteLocationServiceManager* (RLCSM) gehört nicht direkt zur Middleware sondern ist eine zusätzliche, erweiterte Komponente, welche die Menge möglicher nutzbarer LocationServices vergrößert.

Der LCSM erfasst standardmäßig nur lokal auf dem Endgerät installierte LocationServices. Um auch dynamisch entfernte Ortungsdienste auffinden und anbinden zu können wird der RemoteLocationServiceManager benötigt. Er überwacht entfernte Netzwerke und kann dem LocationServiceManager weitere Ortungsdienste anbieten, wenn sie gefunden wurden.

Vielfältige Implementationen sind vorstellbar. Eine Möglichkeit besteht darin, zu überwachen, ob der Nutzer in lokalen Netzwerken angemeldet ist. Wenn ja, kann auf Service-Broadcasts gelauscht werden, ähnlich dem Prinzip von WS-Discovery[40].

Wird ein kompatibler Ortungsdienst gefunden, werden die Eigenschaften abgerufen und ein neuer LCS entsprechend den Schnittstellenvorgaben des LCSM instanziiert, und bekannt gemacht.

Wird dieser Dienst er über die Middleware aufgerufen, werden die Anfragen an den entfernten Dienst weitergeleitet. Statusänderungen können entsprechend erfolgen, je nach dem, ob valide Daten empfangen werden, oder Fehler auftreten wie Abbruch der Netzwerkverbindung.

5.2 Locationmodel

In Kapitel 2.4.1 wurde auf die verschiedenen Eigenschaften von Standortinformationen eingegangen, und welche Daten und Datentypen zu einem Standort vorliegen können. Um diese Informationen verarbeiten und austauschen zu können, muss auf ein gemeinsames Datenmodell zurückgegriffen werden, welches die Daten strukturiert.

Wie erwähnt wurde, kann man grundsätzlich unterscheiden in geografischen Standort (Koordinaten) und symbolischen Standort (Bezeichnungen). Dabei können verschiedene Referenzsysteme oder Ontologien verwendet werden zum Beispiel WGS-84[41] für Koordinaten oder Postanschrift für symbolischen Standort. Neben den eigentlichen Standortinformationen sollte daher auch die Information über verwendetes Referenzsystem oder Ontologie enthalten sein.

Ein einfaches, allgemeines und flexibles Model könnte so aussehen wie in Abbildung 5.3 dargestellt wird.

Ein Location-Objekt enthält mindestens eines oder alle der drei Haupformate geografische Daten (*Coordinates*), symbolische Daten (*Symbolic*) oder Bewegungsdaten (*Movement*). Das geografische

[40] Web Services Dynamic Discovery, ein Discovery Protocol zum Finden von Web Services, aktuelle Spezifikation 1.1 http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.html (Stand: 27.06.2010)
[41] World Geodetic System 1984 [wgs00]

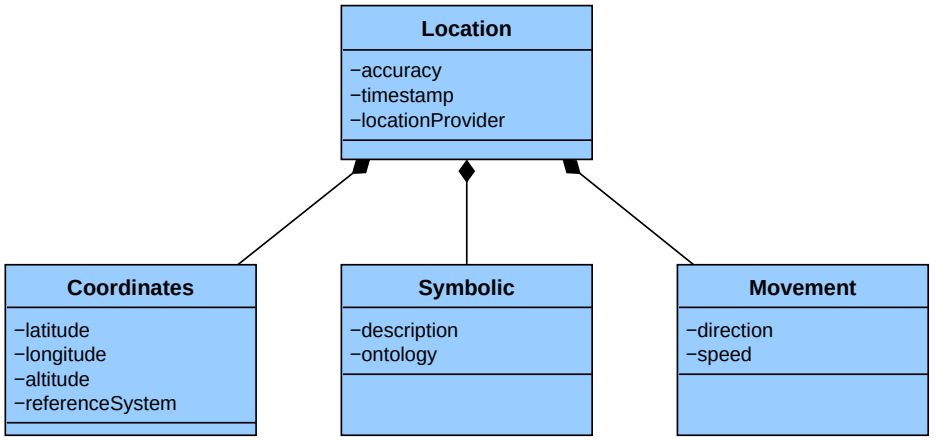


Abb. 5.3: Einfaches Modell für Standortinformationen

Datum enthält das Tupel aus Länge (*Latitude*), Breite (*Longitude*), Höhe (*Altitude*) und eine Angabe zum verwendeten Referenzsystem (*ReferenceSystem*). Das symbolische Datum enthält eine Beschreibung des symbolischen Standorts *(Description*) und eine Angabe zur verwendeten Beschreibungsart bzw. Ontologie (*Ontology*). Die Beschreibung kann als einfacher, eindeutiger Bezeichner, z. B.: "Brandenburger Tor", einfache textliche Beschreibung, z. B.: "Deutschland,10117 Berlin-Mitte, Pariser Platz", oder formatierter Text z. B.: in XML vorliegen, je nach verwendeter Ontologie. Die Bewegungsdaten bestehen aus Geschwindigkeit (*Speed*) und Bewegungsrichtung (*Direction*).
Weitere Daten sind ein Zeitstempel (*Timestamp*) der Standortermittlung, Genauigkeit (*Accuracy*) und Name des Ortungsdienstes, der die Ermittlung durchgeführt hat (*LocationProvider*).

5.3 Abläufe und Vorgänge

Im Folgenden sollen die verschiedenen Abläufe und Vorgänge erläutert werden, die durch das Zusammenspiel der Komponenten entstehen und von der Ortungsdmiddleware, oder genauer gesagt dem LocationEstimator durchgeführt und gesteuert werden müssen. Abbildung 5.4 skizziert den Gesamtablauf.

Die Bearbeitung einer Anfrage beginnt mit der Initiierungsphase (*Init*), in welcher die funktionalen und nichtfunktionalen Anforderungen ausgewertet werden. Anhand des Typs (single, periodic, proactive) entscheidet sich das weitere Vorgehen, welche Aufgaben wie erledigt werden sollen und wann eine Anfrage beendet ist.

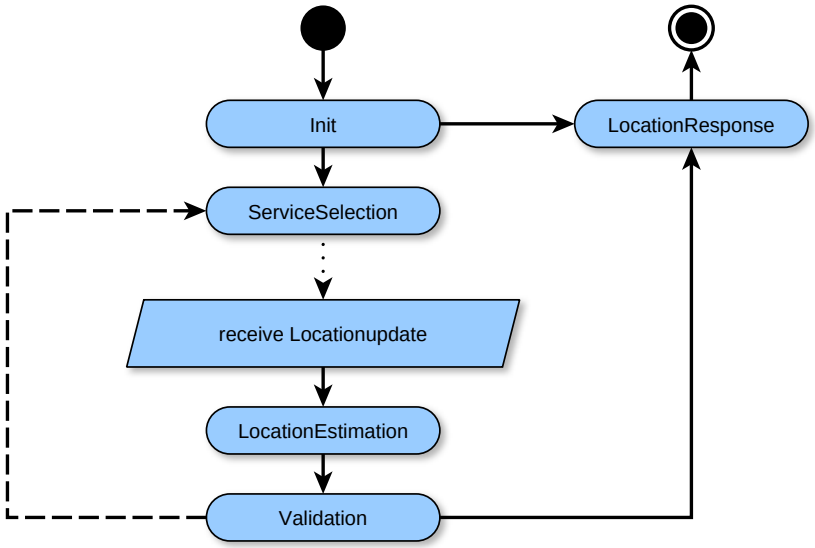


Abb. 5.4: Gesamtablauf der Standortbestimmung durch die Middleware

Zunächst wird geprüft ob bereits Standortinformationen zur Verfügung stehen und ob sie die Anforderungen erfüllen. Anschließend wird das erste *LocationResponse* gesendet.

Die Vorgänge zur Ermittlung neuer Standortdaten orientieren sich an den in Kapitel 4.1.2 ermittelten Schritten. Der LCSM ist für das Auffinden verfügbarer Ortungsdienste verantwortlich. Aus dieser Menge werden anhand der Anforderungen ein oder mehrere Dienste ausgewählt (*ServiceSelection*) und aktiviert.
Treffen schließlich Standortdaten (*Locationupdates*) ein, werden sie in der Validierungsphase (*Validation*) ausgewertet. Dabei wird geprüft, ob der jeweilige Ortungsdienst die in der Dienstbeschreibung angegebenen Paramter auch erfüllt und zuverlässig arbeitet oder möglicherweise durch einen anderen Dienst ersetzt werden sollte.

Parallel wird die middelwareinterne Standortbestimmung (*LocationEstimation*) durchgeführt. Wenn gefordert, wird ein neues Locationupdate in ein anderes Format transformiert. Außerdem wird eine Aggregation zwischen dem zuletzt eingetroffenen Update und kürzlich eingetroffener Updates anderer, möglicherweise gleichzeitig aktivierter, Dienste durchgeführt. Ergebnis ist ein neues Basisstandortdatum (*Basiclocation*).
Dieses wird anschließend überprüft, ob es die Anforderungen der Anfrage erfüllt bevor es mit Statusinformationen an den anfragenden LBS übertragen wird.

Einzelne und periodische Anfragen unterscheiden sich bei der Bearbeitung nur geringfügig. Werden in der Initiierung bereits gültige Standortinformationen gefunden, ist die Einzelanfrage durch

Übermittlung dieser Daten und Statusinformationen bereits abgeschlossen. Ansonsten erfolgt wie bei periodischen Anfragen die Anstoßung des Bestimmungsprozesses. Während aber bei der periodischen Anfrage kontinuierlich Locationupdates empfangen, verarbeitet, ausgewertet, und an LBS gesendet werden, endet der Vorgang bei einer Einzelanfrage sobald die Qualitätsanforderungen das erste Mal erfüllt wurden.

5.3.1 Statusinformationen

Um den qualitätsbewussten Ansatz umzusetzen, und LBS die Möglichkeit zu geben Informationen über den aktuellen Stand zu erhalten gibt es Statusinformationen. Sie beziehen sich auf zwei Aspekte: die eigentlichen Standortinformationen und den Ermittlungsvorgang.

Der *LocationStatus* gibt Auskunft über den aktuellen Stand der ermittelten Standortinformationen bezogen auf die Anforderungen des LBS. Drei Fälle werden unterschieden:

noLocation es stehen (noch) keine Standortdaten (im geforderten Format) zur Verfügung

invalidLocation es stehen zwar Standortdaten zur Verfügung, sie erfüllen aber nicht die Anforderungen des LBS, zum Beispiel, weil die Genauigkeit nicht ausreicht oder die Bestimmung bereits zu lang zurückliegt.

validLocation es stehen gültige Standortdaten zur Verfügung, welche den Anforderungen des LBS entsprechen.

Der *OperationStatus* gibt Auskunft über den Fortschritt der Standortermittlung.

inProgress Standortermittlung wird durchgeführt, es liegen aber noch keine, den Anforderungen entsprechende Ergebnisse vor.

finished Der Vorgang der Standortermittlung ist abgeschlossen. Im positiven Fall wurde er erfolgreich durchgeführt und gültige (validLocation) oder ungültige (invalidLocation) Daten übermittelt. Im negativen Fall wurde der Versuch abgebrochen, da zur Zeit keine Ortungsdienste verfügbar sind, welche die geforderten Daten liefern können oder ein Fehler aufgetreten ist.

Vor allem wenn die Ermittlung längere Zeit in Anspruch nimmt, ist der LBS zeitnah über diese Verzögerungen informiert. Erfolgt zum Beispiel eine einfache Anfrage, kann zunächst eine Antwort mit momentan ungenauen Standortdaten geschickt werden. Der LBS kann dann entscheiden, ob er diese Daten verarbeitet, und anzeigt, oder den Nutzer auf die Verzögerung hinweißt oder gänzlich ignoriert und einfach nur wartet bis die genaueren Standortdaten ermittelt wurden und übertragen werden.

5.3.2 Dienstauswahl

Der Vorgang der Dienstauswahl (*ServiceSelection*) ist dafür verantwortlich, aus der Menge verfügbarer Ortungsdienste einen oder mehrere auszuwählen und über den LCSM zu aktivieren bzw. zu deaktivieren. Die Herausforderung besteht darin, die "richtigen" Dienste auszuwählen. Was richtig bedeutet, und wie schließlich die Auswahl erfolgt, kann durch ganz unterschiedliche Strategien festgelegt werden. Die Auswahl sollte daher als austauschbar zu implementierenden Vorgang realisiert werden.

Ein trivialer Ansatz wäre, aus der Menge verfügbarer Dienste, immer den besten auszuwählen, zum Beispiel den mit der höchsten Genauigkeit. Eine gänzlich anderer, trivialer Ansatz wäre, einfach alle verfügbaren Dienste gleichzeitig auswählen, um nach Aggregation der verschiedenen Locationupdates ein Maximum an Genauigkeit erreichen zu können.

Der Mehrwehr der Middleware kommt jedoch erst zum tragen, wenn die Auswahl "intelligent" erfolgt, in dem die Anforderungen des LBS mit den Möglichkeiten verfügbarer Ortungsdienste abgeglichen werden. In die Auswahl fließen dann verschiedene Kriterien ein, wie das geforderte Standortformat, Intervall und Genauigkeit sowieso die statischen und zur Laufzeit ermittelten dynamischen Eigenschaften der Ortungsdienste wie lieferbare oder transformierbare Formate, Genauigkeit, Latenz und Kosten.

Eine *optimale* Strategie versucht die Menge der aktivierten Ortungsdienste so zu wählen, dass die gefordere LBS-Qualität gerade erfüllt wird, jedoch mit niedrigstem Aufwand an Kosten. Solch ein komplexer Vorgang hängt von vielen Faktoren und unterschiedlichen Bewertungen ab, weshalb es *die* optimale Lösung nicht gibt. Vielmehr sind ganz verschiedene Herangehensweisen und Lösungen denkbar, von denen im Folgenden ein möglicher Ansatz näher vorgestellt werden soll.

Außerdem muss beachtet werden, dass Ortungsdienste dynamisch kommen und gehen. Sie können beispielsweise vom Nutzer aktiviert oder deaktiviert werden. Oder in einer Umgebung können neue unbekannte Dienste erkannt und verfügbar werden. In diesem Fall muss die Dienstauswahl aktualisiert werden.

Zusammengefasst muss die Auswahl angepasst werden wenn folgende Ereignisse eintreten:

- Eine neue LBS-Anfrage wurde erstellt.
- Eine LBS-Anfrage wurde beendet.
- Ein gewählter Dienst liefert schlechte Qualität und soll ersetzt werden.
- Die Verfügbarkeit eines Dienstes hat sich geändert (wird verfügbar oder wird nicht verfügbar)

5.3.2.1 Ein Ansatz

Der folgende Ansatz versucht die Menge verfügbarer potentiell nutzbarer LCS nach einem Algorithmus der die Anforderungenkritieren beachtet in eine sortierte Liste zu bringen. Der

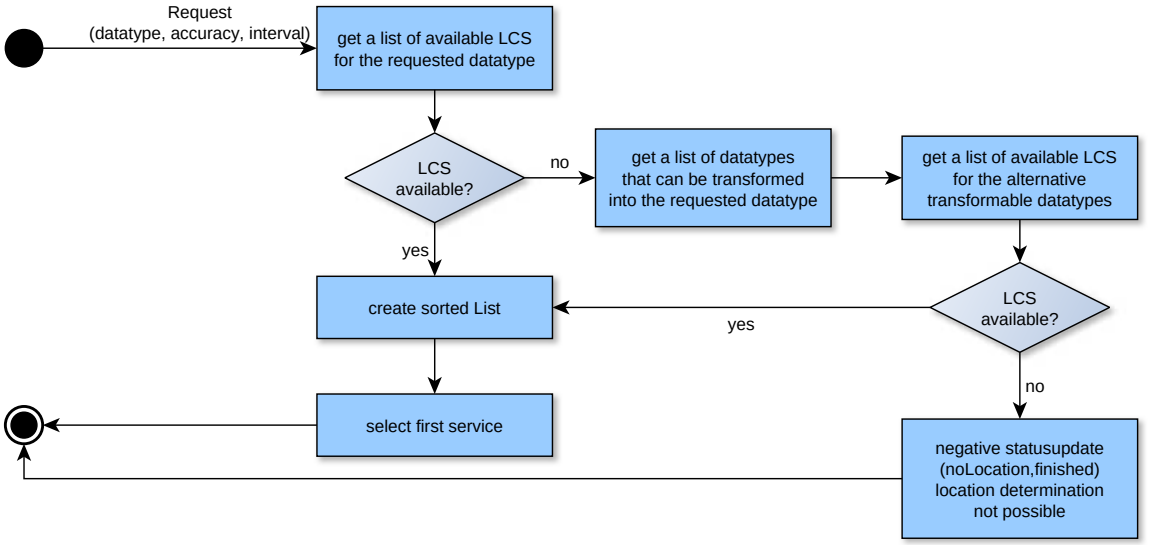


Abb. 5.5: Initiierungsphase eines Ansatzes zur Dienstauswahl

bestgeeignetste LCS steht in der Liste ganz vorn, der am schlechtesten geeignete ganz am Ende. Das hat den Vorteil, dass einfach der erste Service ausgewählt werden kann. Liefert dieser nicht die erwartete Qualität oder wird nicht verfügbar, kann einfach der in der Liste als nächstes stehende Dienst gewählt werden und so weiter, ohne alle Dienste erneut auswerten zu müssen. Ein weiterer Vorteil ist, wird ein vormals nicht verfügbarer Dienst wieder verfügbar, kann sich in der Liste recht einfach wieder zurück gearbeitet werden.

In Abbildung 5.5 wird der initiale Ablauf dargestellt.

Erstellung der sortierten Liste und erste Auswahl Bei einem neuen Request wird zuerst vom LCSM eine Liste verfügbarer Ortungsdienste abgefragt, welche das geforderte Standortformat liefern können. Werden keine Ortungsdienste gefunden, welche das geforderte Format direkt unterstützen, müssen Dienste gefunden werden, deren Locationupdates in das Zielformat transformiert werden können. Der DataManager liefert die Information, für welche *"transformable"* Formate er Methoden vorhält, die sie in das Zielformat transformieren können. Damit wird beim LCSM eine Liste alternativer Ortungsdienste abgefragt.

Die aktuelle Liste wird nun nach einem speziellen Algorithmus sortiert. Dies geschieht in verschiedenen Stufen, jeweils nach einem Anforderungskriterium. Zuerst wird die Liste aufsteigend nach Genauigkeit sortiert. Anschließend wird die Liste durchlaufen und geprüft ob die aktuelle Genauigkeit die Anforderung erfüllt. An der Stelle wo dies nicht mehr gegeben ist wir die Liste

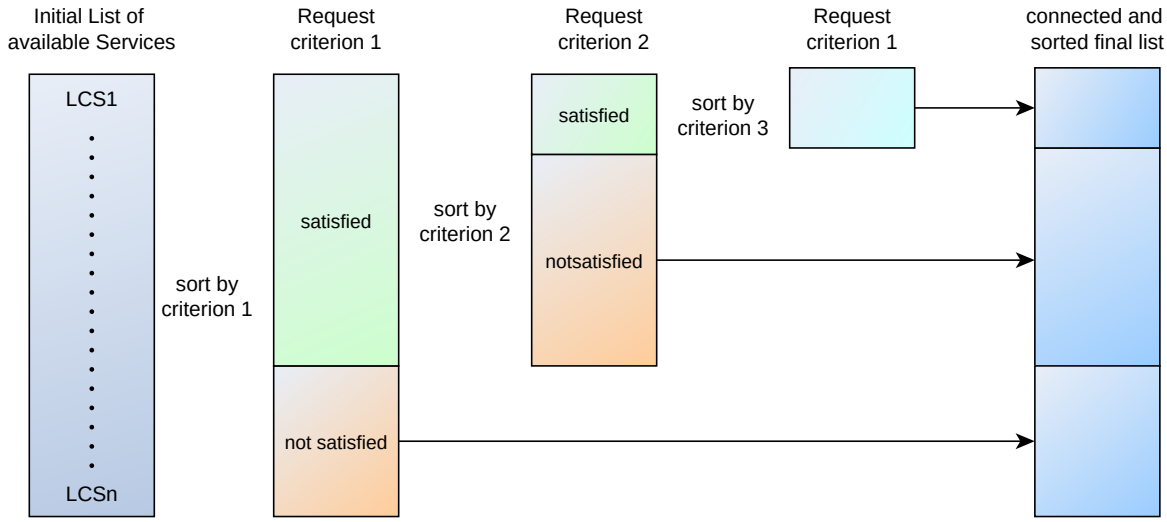


Abb. 5.6: Entwicklung einer ausbalancierten Liste anhand verschiedener Kriterien

aufgeteilt. Der hintere Teil, dessen Einträge alle die Anforderungen nicht erfüllen wird belassen. Der vordere Teil wird mit dem nächsten Kriterium weiterverarbeitet. Es wird aufsteigend nach Intervall sortiert. Anschließend wird wieder die Position gesucht, welche die Anforderung nicht erfüllt und die Liste abermals geteilt. Der hintere Teil verbleibt wieder, und der erste wird für das nächste Kriterium untersucht, zum Beispiel nach Kosten.

Am Ende werden die Teillisten wieder zusammengefügt, in ihrer Teilsortierung jedoch belassen. An erster Stelle steht jetzt ein Dienst, welcher die Ganauigkeit erfüllt, das Intervall erfüllt aber von diesen die niedrigsten Kosten hat. Abbildung ... zeigt anhand von Beispielwerten wie sich eine solche Liste bei verschiedenen Kriterien bilden könnte.

Können weder Dienste gefunden werden, die das geforderte Format direkt oder indirekt liefern, was bedeutet, dass keine Standortbestimmung durchgeführt werden kann, wird der Vorgang mit einer negativen Locationresponse (noLocation, finished) beendet.

Bei der Auswertung der Kriterien werden die statischen Werte aus der Dienstbeschreibung verwendet. Erst wenn ein Dienst tatsächlich ausgewählt und gestartet wurde, sind Daten verfügbar ob zum Beispiel die Genauigkeit auch tatsächlich erreicht wird. Diese Prüfung wird in der Validierungsphase durchgeführt, in deren Ergebnis auch Dienste abgewählt werden können. Diese dürfen im Auswahlalgorithmus dann logischerweise nicht mehr mit betrachtet werden.

Es stellt sich natürlich die Frage, welche Kriterien, in welcher Reihenfolge betrachtet werden.

Statt diese Reihenfolge vorzugeben, ist es auch denkbar, dem LBS die Entscheidung zu überlassen und die Abarbeitung des Algorithmus auf Basis dieser Vorgabe durchzuführen.

Eine weitere Frage stellt sich wie die Behandlung von Diensten mit direkter Formatunterstützung gegenüber Diensten mit indirekter (transformierbarer) Formatunterstützung erfolgt. Ist es sinnvoll, zunächst nur direkte Dienste zu betrachten, und erst wenn diese Option ausgereizt ist, die Auswahl auf indirekte Dienste auszuweiten? Oder sollten direkte und indirekte Dienste gemischt werden? Problematisch dabei könnte sein, dass sich bei einer indirekten Formatunterstützung nicht alle Kriterien ermitteln und bewerten lassen.

Ereignis Statusänderungen Der Auswahlalgorithmus muss auf die Statusänderungen reagieren. Zwei Fälle sind im vorliegenden Ansatz zu betrachten:

- Ein ausgewählter Dienst wird nicht verfügbar, zum Beispiel weil er vom Nutzer deaktiviert wurde. Der Dienst muss abgewählt und deaktiviert werden. Außerdem muss der nächste verfügbare Dienst in der sortierten Liste gefunden und ausgewählt werden.
- Ein Dienst wird verfügbar. Befindet er sich in der sortierten Liste höher, als der derzeit ausgewählte Dienst, dann muss der aktuelle Dienst abgewählt und deaktiviert werden, und der neue Dienst muss ausgewählt und aktiviert werden.
- Der LCSM erkennt einen neuen kompatiblen Dienst. Die Liste muss um diesen erweitert, und der Sortierungsalgorithmus neu durchlaufen werden. Die Dienstauswahl muss aktualisiert werden.

5.3.3 Standortbestimmung

Die Standortbestimmung (*LocationEstimation*) ist der Vorgang, welcher die erhaltenen Standortdaten der verschiedenen LCS wenn gefordert umwandelt und schließlich auswertet und zusammenführt, sei es durch Ergänzung verschiedener Daten (symbolische Daten und Koordinaten nebeneinander) oder Verschmelzung gleichartiger Daten. Sobald ein neues Locationupdate von einem der gewählten LCS eintrifft, muss dieser Schritt durchgeführt werden.

Über den DataManager werden die Standortdaten jedes LCS gespeichert, mindestens des letzten Updates, besser aber auch eine bestimmte Anzahl zurückliegender Updates. Liefert ein LCS das gewünschte Format nicht direkt, muss über eine vom DataManager bereitgestellte Methode eine Transformation in das Zielformat durchgeführt werden.
Schließlich werden die Daten der einzelnen Services aggregiert und der primäre Standort ermittelt (*BasicLocation*). Dies kann durch verschiedene, auswechselbare Strategien erfolgen.

Ist die Bestimmung abgeschlossen und ein neue BasicLocation liegt vor, erfolgt die Validierung, bevor bestehenden LBS-Anfragen mit einer LocationResponse beantwortet werden.

Einfluss auf die Aggregation haben die Angaben zur Genauigkeit, Zeitpunkt sowie die Informationen aus den Dienstbeschreibungen. Die folgenden Betrachtungen und vorgeschlagenen Ansätze zur Aggregation beziehen sich auf das Koordinatenformat.

Gegeben ist eine Menge von LocationUpdates $U_i = (T_i, A_i, L_i)$ von N LocationServices S_i, $1 \leq i \leq N$. Dabei ist L_i die ermittelte Position (Koordinaten), T_i der Zeitpunkt der Bestimmung und A_i die Genauigkeit.

5.3.3.1 Aggregation durch Auswahl

Eine Möglichkeit der Aggregation besteht darin, aus der Menge der (gleichartigen) Locationupdates das mit der höchsten Qualität auszuwählen, und alle anderen zu verwerfen. Zur Ermittlung der höchsten Qualität lässt sich zum Beispiel der Bestimmungszeitpunkt und die angegebene Genauigkeit heranziehen.

Zunächst werden alle Updates verworfen, deren Bestimmungszeitpunkt zu lang zurückliegt: $T_{jetzt} - T_i > t_{max}$. Anschließend wird aus den verbleibenden Updates genau das mit der höchsten Genauigkeit (kleinster Wert) ausgewählt und als neue BasicLocation gespeichert. $U_{basic} = U_j$mit $min(A_j)$.

5.3.3.2 Aggregation durch Fusion

Eine andere Möglichkeit der Aggregation besteht darin, die verschiedenen Locationupdates zu verschmelzen. Aus einer Menge von Koordinaten wird also ein neuer Satz Koordinaten berechnet. Das kann zum Beispiel durch Bildung des gewichteten Mittelwertes (auch Durchschnitt oder Average Score genannt) erfolgen.

Das gewichtete arithmetische Mittel $\bar{x}$ einer Reihe von n Werten x_i und ihren Gewichten w_i ist wie folgt definiert:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i * w_i}{\sum_{i=1}^{n} w_i}$$

Die Koordinaten $C_i = (lat, lon, alt)$ eines LocationUpdates L_i erstrecken sich über die drei Dimensionen Länge (*longitude*), Breite (*latitude*) und Höhe (*altitude*). Der Mittelwert kann für jede Dimension separat berechnet werden. Da nicht jeder Ortungsdienst in der Lage ist, eine Höhenangabe zu liefern, kann bei solchen Updates auch kein Höhenwert mit einberechnet werden. Es gehen dann nur Länge und Breite in die Berechnung ein.

Da verschiedene Locationupdates ganz unterschiedliche Qualität aufweisen können, sollte dementsprechend gewichtet werden, so dass Updates schlechter Qualität weniger stark in die Berechnung

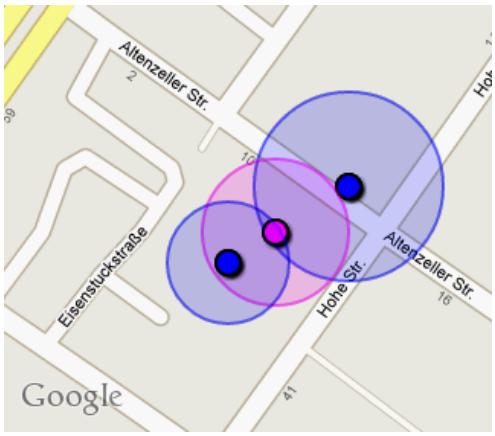


Abb. 5.7: Verschmelzung von Koordinaten durch gewichtete Mittelwertbildung

eingehen, als Updates hoher Qualität. Dabei können wieder die Parameter Genauigkeit und Aktualität zur Ermittlung der Gewichte herangezogen werden. Reicht diese Wichtung nicht aus, müssen weitere Parameter wie Zuverlässigkeitsinformationen über die LocationServices betrachtet werden.

Damit ergibt sich die gemittelte neue Koordinate C_d einer Dimension d durch Mittelwertbildung der n vorhandenen Koordinaten l_{id} und des entsprechenden Gewichts w_i und eines Normierungsfaktors f wie folgt:

$$C_d = f * \frac{\sum_{i=1}^{n} l_{id} * w_i}{\sum_{i=1}^{n} w_i}$$

Eine wichige Frage ist, wie die Genauigkeit des fusionierten Standortes zu bewerten ist. Aus Unklarheit könnte der Wert der schlechtesten Genauigkeit übernommen werden. Möglicherweise entspricht die Genauigkeit des fusionierten Standortes aber ebenfalls dem Durchschnitt der Einzelgenauigkeiten. Abbildung 5.7zeigt das Ergebnis (rot) einer Fusion von zwei Locationupdates (blau), inklusive der Genauigkeitsumkreise.

Koordinaten beziehen sich auf einen Körper, genauer gesagt ein Referenzellipsoid. Berechnungen mit Koordinaten können daher nicht wie in einer Ebene durchgeführt werden. Will man zum Beispiel den Abstand der Koordinaten von Dresden und Hamburg berechnen, und würde man dies mit Geometrie der Ebene durchführen, würde man nicht die Länge des Weges auf der Erdoberfläche von Dresden nach Hamburg berechnen, sondern die direkte Verbindung mitten durch. Für korrekte Berechnungen sind kompliziertere Formeln notwendig, welche die Krümmung des Referenzellipsoides berücksichtigen.

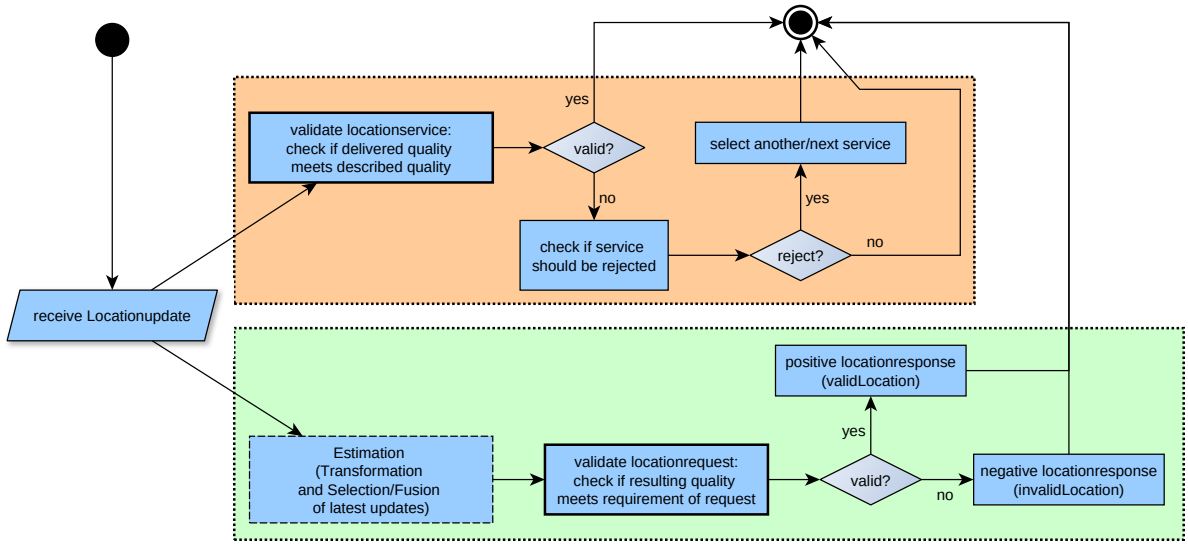


Abb. 5.8: Validierung von Ortungsdienst und Locationupdates

Da sich die zu fusionierenden LocationUpdates allerdings alle in einem erwartungsgemäß kleinen Gebiet befinden, von maximal einigen Kilometern, lässt sich die Erdkrümmung für die Durchschnittsberechnung vernachlässigen.

5.3.4 Validierung

Nachdem ein Ortungsdienst neue Standortdaten übermittelt hat müssen diese verarbeitet und bewertet (*Validation*) werden, wie in Abbildung 5.8 dargestellt wird. Dabei werden zwei Aspekte betrachtet. Zum einen, ob der Dienst die beschriebenen Eigenschaften auch erfüllt (oranger Bereich) und zum anderen, ob die resultierenden Standortinformationen die Anforderungen des LBS erfüllen (grüner Bereich).

5.3.4.1 Qualitätszustand der Ortungsdienste

Die tatsächlich gelieferte Qualität eines Ortungsdienstes wird mit der laut Dienstbeschreibung potentiell möglichen Qualität gegenübergestellt. Auf Grund dieser Bewertung kann dann entschieden werden, ob er weiterhin ausgewählt bleibt, oder abgelehnt wird. Dazu wird in vier Zustände unterschieden:

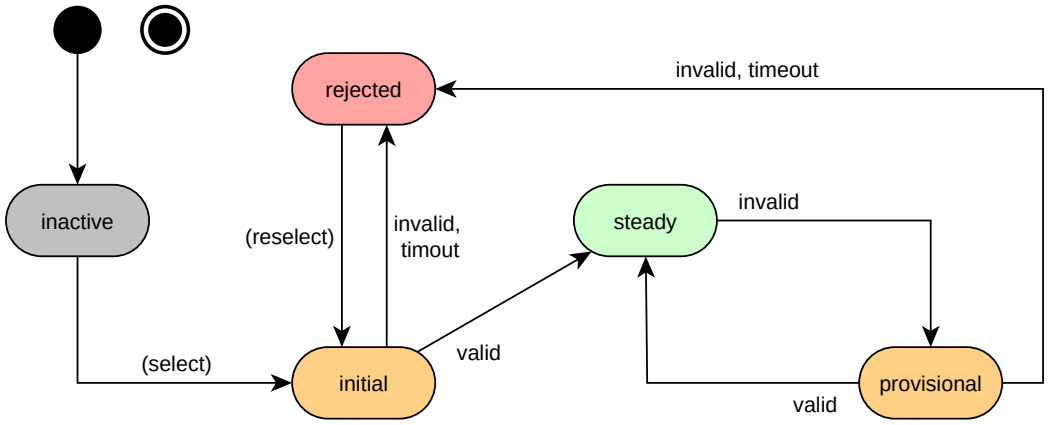


Abb. 5.9: Aus Dienstqualität resultierende Zustände

initial bedeutet, dass der Dienst erst kürzlich gestartet wurde, und noch keinen "stabilen" Zustand erreicht hat, in dem er sein Qualitätspotential ausschöpfen kann.

provisonal bedeutet, dass die Qualität der zuletzt gelieferten Standortdaten unter ein bestimmtes Limit gefallen ist. Sollte die Qualität innerhalb eines bestimmten Zeitintervalls nicht wieder steigen ist dies ein Indiz, dass der Dienst deaktiviert werden sollte.

steady bedeutet, dass die Qualität der zuletzt gelieferten Standortdaten den Anforderungen entspricht und keine Änderungen in der Dienstauswahl getroffen werden müssen.

rejected die Qualität ist innerhalb des gewählten Zeitintervalls nicht gestiegen, so dass der Dienst deaktiviert werden sollte und eine Alternative gefunden werden muss.

Da es Ortungsdienste gibt, die eine gewisse Initiierungsphase benötigen, bis die typische und geforderte Qualität erreicht ist, wird bei Erstaktivierung der Zustand *initial* gesetzt. So wird verhindert, dass beim ersten Locationupdate der Dienst sofort wieder deaktiviert wird, wenn er die Qualität noch nicht erreicht ist. Ob und wie lang diese Phase dauert ist für jeden Dienst unterschiedlich und sollte in der Dienstbeschreibung enthalten sein.

Wird schließlich eine stabile Qualität erreicht, wechselt der Zustand zu *steady*.

Fällt die Qualität nach langer stabiler Zeit unter ein Limit, sollte der Dienst ebenfalls nicht sofort deaktiviert werden, sondern eine bestimmte Dauer überwacht werden, um die Chance zu haben wieder stabile Qualität zu erreichen. Dazu wird er in den Zustand *provisional* versetzt. Die Dauer, die ein Dienst in diesem Zustand bleiben darf, kann von der Middleware und für unterschiedliche Dienste durchaus verschieden vorgegeben werden. So wird eine Fluktuation bei kurzzeitigen Qualitätsschwankungen vermieden.

Ist die Zeit abgelaufen, wird der Dienst mit dem Zustand *rejected* abgelehnt. Dies ist das Signal für den LocationEstimator den Dienst zu deaktivieren und eine neue Dienstauswahl anzustoßen. Dienste im Zustand rejected werden bei einem neuen Auswahldurchlauf übersprungen.

5.3.4.2 Erfüllung der LBS-Qualitätsanforderungen

Nachdem der Vorgang der LocationEstimation durchgeführt wurde, und nach optionaler Transformation und Aggregation die BasicLocation bereit steht, muss die ermittelte Qualität mit den Anforderungen des LBS abgeglichen werden. Daraus ergibt sich der aktualisierte LocationStatus. Sind die Anforderungen erfüllt (validLocation) und es handelt sich um eine Einzelanfrage, wird der operationStatus auf finished gesetzt.
Anschließend wird die LocationResponse gesendet.

5.4 Weitere Merkmale und Konzepte

Im Folgenden sollen noch einige Ideen und Konzepte vorgestellt werden, die in der anschließenden Implementation keine Berücksichtigung fanden, aber trotzdem erwähnenswert sind.

5.4.1 Privatsphäre

Wie bereits erwähnt, stellen die Standortdaten eines Nutzers sensitive Informationen dar, die nur unter Wissen und Kontrolle des Nutzers an LBS weitergegeben werden dürfen. Im Konzept einer Ortungsmiddleware muss dieser Punkt bedacht werden.

Jeder LBS, welcher über die Middleware Standortinformationen abfragen will, muss daher zunächst vom Nutzer autorisiert werden. Beim ersten Zugriff eines LBS sollte dem Nutzer ein Dialog gezeigt werden, in dem er gefragt wird, ob seine Standortdaten an den LBS übermittelt werden dürfen. Der Nutzer hat die Möglichkeit seine Entscheidung zu speichern, so dass bei weiterem Zugriff dieses LBS keine weitere Zwischenfrage gestellt wird. Über Präferenzen der Middleware kann der Nutzer die Zugriffsrechte von LBS verwalten und Zugriff sperren oder freigeben.

5.4.2 Authentifizierung

In Kapitel 4.2.6wurde kurz auf wirtschaftliche Aspekte eingegangen, und die Beziehungen, welche oft zwischen LBS-Anbietern und Standortanbietern bestehen. Damit ein LBS den Dienst eines Standortanbieters nutzen kann erhält er bei einer Vertragsbeziehung oft einen API-Key welcher er bei ansteuern der Schnittstelle des Anbieters übermitteln muss.

Um die Nutzung solche Ortungslösungen in der Ortungsmiddleware zu unterstützen könnte man es ermöglichen, diese Authentifizierungsdaten durchzuschleifen. Werden bei der Dienstauswahl Dienste gefunden und präferiert, welche Authentifizierung benötigen, kann bei der Anfragenden LBS nach verfügbaren Authentifizierungsdaten gefragt werden. Erst wenn diese zur Verfügung gestellt werden, kann der entsprechende Dienst genutzt werden.

6 Implementierung

Im folgenden Kapitel wird die prototypische Implementierung *Quality Aware Location Middleware* (kurz Qalmw) des konzeptionierten Systems erläutert. Es wird detailliert auf die einzelnen Komponenten und die verwendeten Technologien eingegangen.

Die besondere Herausforderung bestand dabei in der Umsetzung einer Lösung für ein modernes Smartphone. Bei Untersuchung bestehender Middlewareansätze wurde festgestellt, dass Umsetzungen nicht immer für die Nutzung auf Smartphones hin entwickelt und getestet wurden, sondern "nur" auf wesentlich leistungstärkeren Notebooks. Die Entwicklung für Smartphones erfordert jedoch eine spezielle Herangehensweise, da wesentlich beschränktere Ressourcen und spezielle Lebenszyklen von Anwendungskomponenten beachtet werden müssen.

Der Fokus des Prototyps liegt auf einer erweiterbaren Umsetzung der Gesamtarchitektur des Konzepts und der losen Kopplung zwischen Middleware und Ortungsdiensten. Es soll ermöglicht werden, neue Ortungsdienste hinzuzufügen zu können, welche von der Middleware ohne weitere Änderungen erkannt werden.
Der Prototyp soll die im Konzept vorgestellten Abläufe realisieren und verschiedene Estimations- und Auswahlalgorithmen unterstützen. Als Beispiele sollen die vorgestellten Estimationsalgorithmen (Vergleich Kapitel 5.3.3) und der Ansatz zur Dienstauswahl (Vergleich Kapitel 5.7) integriert werden. Um das Verhalten der Middleware testen und ermittelte Standortdaten auswerten zu können, soll es eine einfache und funktionale Nutzerschnittstelle geben, in welcher verschiedene Requestparameter spezifiziert werden können, und Locationresponses der Middleware dargestellt werden.

6.1 Technologie

Zunächst wird auf die im Prototyp verwendeten Technologien und deren Eigenschaften und Anforderungen eingegangen.

6.1.1 Endgerät HTC Dream

Als Endgerät wurde ein modernes Smartphone ausgewählt, das HTC Dream[42], welches in Deutschland als T-Mobile G1 vermarktet wird und als erstes Smartphone mit Googles Betriebssystem Android ausgestattet wurde. Es verfügt über moderne Kommunikationsschnittstellen, u. A. GSM Quadband, UMTS, WLAN, Bluetooth; verschiedene Eingebemöglichkeiten wie QWERTZ-Tastatur und kapazitiven Touchscreen und vielfältige Erweiterungen wie den integrierten GPS-Empfänger und Beschleunigungs-, Lage-Sensoren und Kompass. Es eignet es sich daher hervorragend für die Ausführung mobiler LBS und dient als praxistaugliche Referenz.

6.1.2 Entwicklunsplattform Android 1.6

Als Anwendungsplattform wird Android[43] verwendet, das von der Open Handset Alliance[44] entwickelte quelloffene Betriebssystem für mobile Endgeräte in der für das HTC Dream aktuellsten Version 1.6 (vergleich das in Kapitel 4.4.2.3 vorgestellte Android Location Framework).

Die Architektur von Android basiert auf einem Linux-Kernel (aktuell 2.6). Er bildet die Hardwareabstraktionsschicht und kümmert sich um Speicherverwaltung, Prozessverwaltung, und Netzwerkkommunikation.

Kern der Laufzeitumgebung bildet die auf Java-Technik basierende Dalvik Virtual Machine (DVM) und die dazugehörigen Android-Java-Klassenbibliotheken.

Jede Anwendung läuft in einem eigenen Betriebssystemprozess. Er wird von Android gestartet, sobald die Anwendung ausgeführt werden soll, und beendet wenn er nicht länger gebraucht wird, oder Systemressourcen von anderen Anwendungen benötigt werden. Jeder Prozess hat seine eigene DVM, was hohe Sicherheit und Verfügbarkeit garantiert, da sich Anwendungen nicht gegenseitig beeinflussen können.

6.1.2.1 Lebenszyklen und Komponenten

Das zentrale Merkmal der modernen Androidplattform ist das komponentenorientierte Design. Eine Anwendung kann Teile von anderen Anwendungen nutzen (vorausgesetzt diese erlauben es), aber ebenso eigene Teile für die Nutzung in anderen Anwendungen zur Verfügung stellen. Benötigt man in seiner Anwendung zum Beispiel eine geeignete scrollbare Liste für Bilder und eine andere Anwendung hat bereits eine solche Liste implementiert und für andere verfügbar gemacht, kann man einfach diese Komponente aufrufen, anstatt eine solche Liste mühsam selbst zu entwickeln. Das Rad muss also nicht jedes Mal neu erfunden werden.

[42] http://www.htc.com/www/product/g1/overview.html (Stand: 27.06.2010)
[43] http://www.android.com/ (Stand: 27.06.2010)
[44] http://www.openhandsetalliance.com/ (Stand: 27.06.2010)

Die Komponenten einer Anwendung werden in einer strukturierten XML-Datei (*AndroidManifest.xml*), dem Manifest, bekannt gemacht.

Komponenten Im Unterschied zu Anwendungen auf vielen anderen Systemen, bestehen Androidanwendungen aus einzelnen Komponenten, welche vom System bei Bedarf instanziiert und ausgeführt werden. Es gibt vier Typen: *Activities*, *Services*, *Content Provider* und *Broadcast Receiver*, wobei für die Entwicklung des Prototyps und der Beispielanwendung vor allem die ersten beiden von Bedeutung sind.

Eine *Activity* repräsentiert eine grafische Nutzerschnittstelle für eine Interaktion mit dem Nutzer, zum Beispiel eine Liste mit Menüeinträgen, aus denen der Nutzer wählen kann, oder die Darstellung von Photos und zugehörigen Beschreibungen. Eine Anwendung kann aus einer oder mehreren Activities bestehen. Eine davon ist typischerweise als Einstiegspunkt markiert und wird präsentiert, wenn die Anwendung gestartet wird. Bewegung zu weiteren Activities erfolgt in dem die aktuelle Activity die nächste aufruft.

Ein *Service* verfügt über keine grafische Oberfläche sondern läuft im Hintergrund. Populäres Beispiel ist ein Mediaplayer, welcher Lieder aus einer Liste abspielt. Die Playeranwendung würde typischerweise verschiedene Activities haben, um Lieder auszuwählen und die Wiedergabe zu starten. Die Wiedergabe selbst würde vom Service im Hintergrund ablaufen, so dass der Nutzer andere Anwendungen mit anderen Activities ausführen kann während er der Musik lauscht.
Services können an denselben Prozess wie die aufrufende Anwendung und das zugehörige Userinterface gekoppelt sein (sogenannter *local service*) oder in einem eigenen Prozess gestartet werden (*remote service*). Letzterer ist von entscheidender Bedeutung, wenn der Service die Lebensdauer der aufrufenden Komponente übersteigen soll wie im Mediaplayerbeispiel.
Kommuniziert wird mit einem Service über eine von ihm angebotene Schnittstelle. Dazu muss zuerst eine Verbindung (*ServiceConnection*) hergestellt werden, und der Service muss, falls er nicht bereits läuft, gestartet werden.

Ein *Broadcast receiver* empfängt und reagiert auf Broadcast-Meldungen wie zum Beispiel dass der Akku fast leer ist, die Zeitzone gewechselt wurde, oder eine Datenverbindung hergestellt wurde.

Ein *Content provider* einer Anwendung stellt eine bestimmte Menge ihrer Daten anderen Anwendungen zur Verfügung. Daten können beispielsweise im Dateisystem oder einer SQLite[45] Datenbank gespeichert werden.

Lebenszyklen Jede Anwendungskomponente hat einen Lebenszyklus der beginnt, wenn sie von Android instanziiert wird und endet, wenn die Instanz zerstört wird. Dazwischen ist sie manchmal aktiv oder inaktiv und im Fall einer Activity für den Nutzer sichtbar oder unsichtbar. Android

[45] http://www.sqlite.org/ (Stand: 27.06.2010)

versucht Anwendungsprozesse so lang wie möglich zu erhalten, wird aber irgendwann, wenn Speicher knapp wird, ältere Prozesse entfernen müssen. Um zu entscheiden welcher Prozess zu erhalten und welcher zu entfernen ist stuft Android jeden Prozess in eine Wichtigkeitsrangordnung ein.

Während Activities, Content Provider und Broadcast Receiver kurzlebige Komponenten sind, die prinzipiell zu jeder Zeit von Android beendet werden können, stellt der Service eine langlebige Komponente dar.
Wenn das erste mal eine Verbindung (ServiceConnection) zu einem Service hergestellt wird, instanziiert ihn Android. Steht die Verbindung, kann mit dem Service kommuniziert werden. Damit er aktiv bleibt, auch nachdem die letzte Activity ihre Verbindung getrennt hat, muss ein Service explizit gestartet werden. Erst dann ist er im langlebigen Zustand. Entsprechend muss er zum beenden auch explizit gestoppt werden. Verbindung herstellen und trennen kann, während er läuft, beliebig oft und von beliebig vielen anderen Komponenten erfolgen.

Komponentenaufruf Um unabhängige Komponenten untereinander und zum Androidsystem zu verbinden, gibt es einen standardisierten Mechanismus: asynchrone Nachrichten, genannt Intents ("Absichtserkklärungen"). Damit wird eine lose Kopplung der Bestandteile einer Anwendung ermöglicht und die Komponentennutzung und Austauschbarkeit vereinfacht.

Zum Inhalt eines Intents zählen u. A. eine *Action* die ausgeführt werden soll und eine *URI* zu Daten, die dazu genutzt werden sollen, zum Beispiel die Darstellung eines Bildes, oder das Editieren von Text durch den Nutzer.

Es gibt zwei verschiedene Arten von Intents. Bei expliziten Intents muss die Empfängerkomponente bereits bei der Programmierung bekannt und identifiziert sein, typischerweise durch einen Klassennamen. Implizite Intents adressieren keinen Empfänger, sonder überlassen es den Komponenten, darauf zu reagieren. Sie werden ins Leere abgeschickt, mit Hoffnung, dass es Komponenten gibt, die damit etwas anfangen können.

Um zum Beispiel einen Service zu starten wird ein Intentobjekt an die Android Systemmethode *Context.startService* übergeben. Android ruft daraufhin die *onStart*-Methode des adressierten Service auf.

6.1.2.2 Interprozesskommunikation

Der bevorzugte Weg in Android mit Remote Services zu kommunizieren ist Interprozesskommunikation (Inter Process Communication, kurz IPC). Dazu wird eine Schnittstelle via IDL definiert. Die Interface Definition Language ist eine allgemeine Spezifikationssprache, welche Datenaustausch und Kommunikation zwischen Prozessen unabhängig von Programmiersprache

und Betriebssystem ermöglicht. Auch Android selbst nutzt sie. Daten werden in primitive Datentypen zerlegt, die von jeder Anwendung plattformunabhängig verstanden werden können. Googles angepasste Version für Anroid heißt AIDL, Android-IDL

[BP09, Bur09, Mur09]

6.1.3 Ortungslösungen

LocationProvider des Android Location Framework

Auf dem G1 sind bereits zwei Ortungslösungen verfügbar, welche über das Android Location Framework (siehe Kapitel 4.4.2.3) angesprochen werden können: der *GpsLocationProvider* und der *NetworkLocationProvider*[46]. Sie werden unterstützt von und basieren auf den Google Location Services, welche zum Beispiel auch für Google Latitude, oder die Geolocation API in Mozilla Firefox verwendet werden (Siehe 4.4.2.2). Leider gibt es keine Spezifikationen über die genauen Eigenschaften. Es wird lediglich eine Unterscheidung in fein (*fine*, bedeutet exakte Position) und grob (*coarse*, bedeutet ungefähre Position) getroffen.

Der GpsLocationProvider bindet den internen GPS-Empfänger an. Dabei ist hinsichtlich Verfügbarkeit und Nutzung keine manuelle Unterscheidung zwischen normalem GPS oder A-GPS möglich. Aus den Einträgen des Android Logging Systems wird allerdings ersichtlich, dass der GpsLocationProvider das Vorhandensein einer Datenverbindung überwacht, was die Vermutung nahelegt, dass A-GPS möglich ist und verwendet wird, falls eine Datenverbindung besteht.

Die horizontale Genauigkeit kann bis zu 5m betragen, die vertikale Genauigkeit ist unbekannt. Unter guten Bedingungen (ausreichend Sichtverhältnisse zu Satelliten) ist sowohl Bewegungsrichtung als auch Geschwindigkeit verfügbar. Intervalle sind kleiner als im Abstand von einer Sekunde möglich. Die TTFF kann wenige Sekunden bis über eine Minute dauern, abhängig davon, wie lang die letzte Bestimmung zurücklag, wie weit sich seitdem bewegt wurde, und wie gut die Sichtverhältnisse sind. Für genauere Abschätzungen sollten umfangreiche Tests durchgeführt werden.

Der NetworkLocationProvider integriert im Prinzip zwei verschiedene Ortungslösungen: Mobilfunkortung via Cell-ID und WLAN-Ortung via Fingerprintverfahren. Beide basieren auf den Google Location Services, d. h. sie nutzen Cell-ID-Datenbank und WLAN-Radiomap-Datebank von Google. Deswegen ist eine notwendige Bedingung das Vorhandensein einer Datenverbindung, entweder über Mobilfunk oder WLAN.

Je nachdem, welche Verbindungen gerade aktiviert und verfügbar sind, werden entsprechend unterschiedliche Ortungsqualitäten geliefert. Bei Nutzung von Cell-ID beträgt die horizontale Genauigkeit zwischen 500 und 1000m und aufwärts, bei WLAN-Ortung bis zu 100m. Vertikale

[46] http://developer.android.com/reference/android/location/LocationProvider.html (Stand: 27.06.2010)

	Accuracy	Latency (Interval)	Coordinates	Symbolic	Speed	Direction	Charges	Costs
GpsProvider	5	<1000	yes	no	yes	yes	0	3
NetworkProvider (cell−id & wlan)	100	60000	yes	no	no	no	0	2
NetworkProvider (only cell−id)	1000	60000	yes	no	no	no	0	1

Tab. 6.1: Eigenschaften der eingesetzten Ortungslösungen

Ortung, sowie Bewegungsrichtung und -geschwindigkeit sind nicht verfügbar. Die TTFF beträgt nur wenige Sekunden. Abfrageintervalle unter 60 Sekunden sind allerdings nicht möglich.

Durch die fehlenden Qualitätsangaben seitens Google Android, und die intern durch verschiedene Systembedingungen wechselnden Eigenschaften lässt sich eine zuverlässige a priori-Beschreibung der Qualitätsparameter nur unzureichend angeben. Für den Prototyp wurden die in Tabelle 6.1 aufgelisteten Werte angenommen. Die Kosten (costs) ergeben sich aus der Einschätzung über den Energiebedarf im Vergleich untereinander. GPS belastet den Akku am meisten, wurde daher mit drei bewertet. Wird WLAN verwendet wurde der NetworkProvider mit 2 bewertet, ansonsten mit eins. Monetäre Kosten (charges) entstehen bei keiner Lösung.

Zugriff auf einen LocationProvider erhält man, in dem die Android-Schnittstelle *LocationListener* implementiert und beim Android LocationManager[47] registriert. Als Parameter kann man das kleinste Zeitintervall (minTime) oder das kleinste Entfernungsintervall (minDistance) vorgeben.

6.2 Architekturüberblick

Die Architektur des Prototyps orientiert sich stark am Konzept. Viele der dort vorgestellten Komponenten lassen sich direkt wiederfinden.

Die Ausführung der Middleware findet auf drei Ebenen statt. Oben befinden sich die LBS. Sie werden durch die Androidkomponenten Activitiy oder Service implementiert. In der Mitte befindet sich die Middleware, welche durch einen Service implementiert wird und alle wichtigen

[47] http://developer.android.com/reference/android/location/LocationManager.html (Stand: 27.06.2010)

Hauptkomponenten (LE, LCSM, DM) enthält. Unten befinden sich die verschiedenen LCS, welche jeweils eine Ortungslösung kapseln und ebenfalls durch je einen Service implementiert werden. Die Kommunikation zwischen den Ebenen erfolgt über via AIDL definierter Schnittstellen und IPC.

LBS, Middleware und die verschiedenen LocationServices sind prinzipiell weitgehend voneinander unabhängige Anwendungen und würden in der Praxis logischerweise auch in verschiedenen Anwendungspaketen vorliegen. Zur Vereinfachung der Anwendungsentwicklung wurden allerdings alle Teile, natürlich in entsprechend unterschiedlichen packages, in einem gemeinsamen Anwendungspaket zusammengefasst. Die Definition der implementierten Ortungsdienste, der Schnittstelle zur Middleware und der Beispielanwendungen erfolgt demnach in der gemeinsamen AndroidManifest.xml. Eine Trennung wäre aber jederzeit möglich. Es müssten nur die gemeinsam genutzten Komponenten geteilt werden: das Location Model und die Schnittstellenbeschreibungen für die Middleware und die LocationServices.

6.3 Locationmodel

Das Android Location Framework liefert kein vollständiges Locationmodel, sondern mit der Klasse *android.location.Location*[48] nur eine Repräsentation für geografische Standortdaten. Diese enthält die Koordinaten Länge (*latitude*) und Breite (*longitude*), einen Zeitstempel (*timestamp*), den Namen des LocationProviders und optional Höhe (*altitude*), Geschwindigkeit (*speed*) und Richtung (*bearing*).
Da es einige Androidspezifika in der Implementierung gibt, wurde die Android-Location-Klasse nicht in das entwickelte Locationmodel integriert.

Stattdessen wurde eine eigene, einfache Lösung implementiert, entsprechend dem Vorschlag aus Kapitel 5.3, allerdings ohne jeden Standortdatentyp als eigene Klasse zu realisieren.

Im package *de.tud.qalmw.model* befinden sich die zwei Klassen *QLocation* und *Coordinates* wie in Abbildung 6.1 gezeigt wird. Coordinates repräsentiert geografische Standortdaten in Form von Länge (*latitude*), Breite (*longitude*), Höhe (*altitude*) und dem verwendeten Referenzsystem (*referenceSystem*), standardmäßig WGS-84[49]. Die anderen beiden Standorttypen *symbolic* und *movement* werden direkt als Attribute der Klasse QLocation gespeichert. *Symbolic* beschreibt den symbolischen Standort und *symbolicOntology* die verwendete Beschreibungssprache/Referenzsystem. Movement wird durch Geschwindigkeit(*speed)* und Richtung (*direction*) repräsentiert. Genauigkeit (*accuracy*), Zeitstempel (*time*) und der verwendete Locationprovider komplettieren die Klasse.

[48] http://developer.android.com/reference/android/location/Location.html (Stand: 29.06.2010)
[49] World Geodetic System 1984 [wgs00]

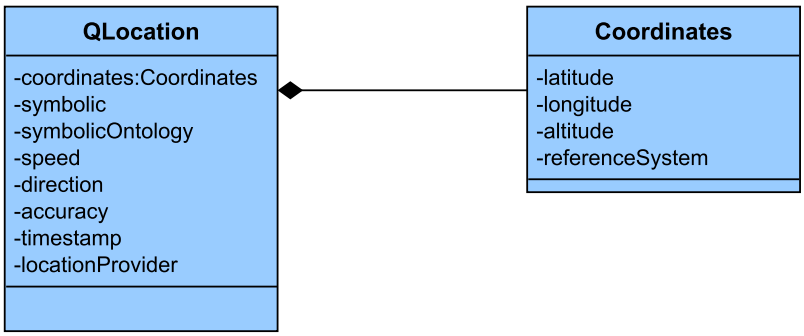


Abb. 6.1: Locationmodel des Prototyps

6.4 Hauptanwendung und LBS-Schnittstelle

Die Hauptanwendung wird durch die Klasse *QalmwLocationService* im package *de.tud.qalmw.service* repräsentiert, welche von der Android-Service-Komponente *android.app.Service*[50] abgeleitet ist.

Innerhalb dieser Klasse werden die Hauptkomponenten der Middleware LocationEstimator, LocationServiceManager und DataManager instanziiert.

Um LBS die Möglichkeit zu geben Anfragen an die Middleware zu stellen, wird die Schnittstelle *IQalmwLocationService* mittels AIDL definiert. Sie bietet Methoden um Anfragen zu erstellen, z. B.: *requestLocationUpdate* und Anfragen zu entfernen z. B.: *removeLocationUpdate*. Dabei muss ein Callback[51] (*ILocationCallback*) übergeben werden, über welches die Middleware die ermittelten Standortdaten und Statusinformationen sendet.

Die Serviceklasse wird automatisch von Android instanziiert, sobald der erste LBS eine Verbindung aufbaut, und beendet, wenn alle Verbindungen getrennt wurden. Damit die Middleware in den langlebigen Zustand kommt, in dem auch Standortdaten ermittelt werden, ohne dass ein LBS verbunden ist, muss der Service gestartet werden. Dies soll jedoch nicht explizit durch LBS erfolgen sondern automatisch. Das Starten (*startService*) und Stoppen (*stopService*) wird durch das Vorhandensein von LocationCallbacks gesteuert. Sobald die erste LBS-Anfrage kommt und ein Callback registriert, startet sich der Service selbst. Sobald die letzte Anfrage und damit das letzte Callback entfernt wurde, stoppt sich der Service wieder selbst. Abbildung 6.3 verdeutlicht den Zyklus.

Damit das Verhalten des Prototyps in einer Testanwendung überwacht und ausgewertet werden kann, lässt sich ein weiteres Callback (*IStatusCallback*) registrieren, über welches die Middleware

[50] http://developer.android.com/reference/android/app/Service.html (Stand: 27.06.2010)
[51] Rückruffunktion, englisch: callback function

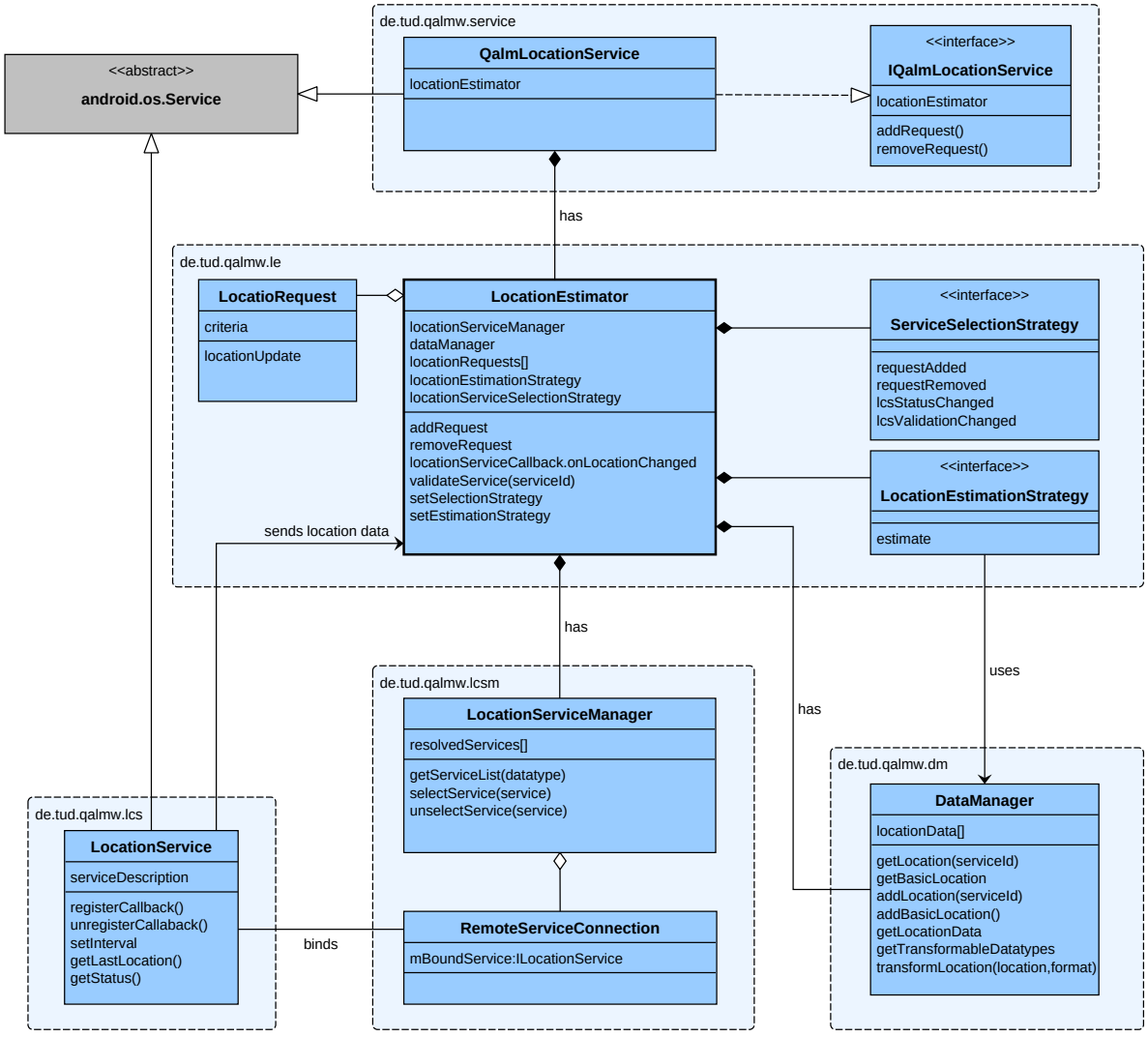


Abb. 6.2: Vereinfachtes Klassendiagramm des Prototyps

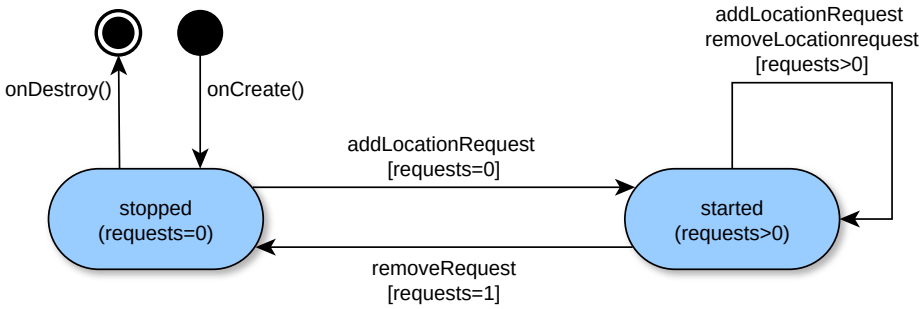


Abb. 6.3: Zustände Middleware

Statusnachrichten in Textform sendet. Diese können dann auf dem Bildschirm des Endgerätes dargestellt werden. Außerdem gibt es Methoden um die Strategien für Estimation und Selection zu wechseln, und die vollständige Datensammlung des DataManagers auszulesen.

6.5 Kernkomponenten: LE und DM

Der Kern der Middleware und die Geschäftslogik befindet sich in der Klasse *LocationEstimator* im package *de.tud.qalmw.le*. Sie wird innerhalb der Serviceklasse QualityAwareLocationMiddleware instanziiert und enthält selbst wiederum die flankierenden Komponenten LocationServiceManager und DataManager.

Wird von einem LBS über die Schnittstelle des Service eine Anfrage gesendet, wird ein neues *LocationRequest*-Objekt erstellt. Es hält das Callback zum senden der LocationResponse vor, speichert die Anforderungen aus der LBS-Anfrage und enthält Logik zum Abgleichen ermittelter Standortdaten mit LBS-Anforderungen und ermitteln der entsprechenden Statusinformationen (*LocationStatus* und *OperationStatus*).

Der DataManager befindet sich im package *de.tud.qalmw.dm* und speichert temporär eingetroffene Locationupdates der verschiedenen genutzten Ortungsdienste und die ermittelte Basiclocation in einer Collection[52]. Er bietet dementsprechend Methoden, neue Updates hinzuzufügen, und gespeicherte Updates auszulesen. Der DataManager ist außerdem für das Bereitstellen von Transformationsmethoden für verschiedene Standortformate verantwortlich, was im Prototyp allerdings nicht implementiert wurde.

[52] Mit Collection ist eine Datenstruktur der Java Collection API gemeint, *java.util.Collection* (für Listen, Mengen, Schlangen) oder *java.util.Map* (für Assoziativspeicher) http://openbook.galileocomputing.de/javainsel8/javainsel_12_001.htm (Stand: 29.06.2010)

Die wichtigste Funktionalität wird durch die beiden Vorgänge Dienstauswahl (ServiceSelection) und Standortbestimmung (LocationEstimation) erbracht (vgl. Kapitel 5.3.2 und 5.3.3). Da die Algorithmen austauschbar und erweiterbar sein sollen, wurde das Entwurfsmuster Strategie[53] verwendet. Um die Algorithmen nutzen zu können wird lokal die jeweils gewünschte Implementation instanziiert.

6.5.1 ServiceSelection

Das Interface *LocationServiceSelectionStrategy* im package *de.tud.qalmw.le.selection* deklariert vier Methoden für die vier Ereignisse, in denen die Dienstauswahl aktualisiert werden muss: bei Erstellung einer neuen Anfrage (*requestAdded*), bei Entfernen einer Anfrage (*requestRemoved*), bei Änderung der Verfügbarkeit eines Ortungsdienstes (*lcsStatusChanged*) und bei Änderung des Qualitätszustandes eines (gewählten) Ortungsdienstes (*lcsValitationChanged*).

Für den Prototyp wurden zwei Ansätze implementiert. Die *SelectAll*-Klasse wählt einfach alle verfügbaren Ortungsdienste aus, die das geforderte Format unterstützen. Die *BalancedSelection*-Klasse setzt den in Kapitel 5.3.2 vorgestellten Ansatz um. Der Schwerpunkt wurde auf das Erstellen der ausbalancierten Liste und die Reaktion auf Statusänderungen der verfügbaren Ortungsdienste gesetzt. Das lcsValidationChanged-Ereignis wird zum aktuellen Zeitpunkt nicht verarbeitet.

Abfrage verfügbarer Services und Aktivierung erfolgt über den LocationServiceManager.

6.5.2 LocationEstimation

Das Interface *LocationEstimationStrategy* im package *de.tud.qalmw.le.estimation* deklariert die Methode *estimate*, welche immer aufgerufen wird, nachdem ein neu eingetroffenes Locationupdate im DataManager gespeichert wurde. Die Estimationsalgorithmen fragen über diesen zuerst die letzten Locationupdates ab und speichern nach Ermittlung die neue Basiclocation.

Für den Prototyp wurden die in Kapitel vorgestellten Ansätze implementiert. Die *SelectBestAccuracy*-Klasse filtert zunächst zu alte Standortdaten heraus und wählt anschließend das Update mit der höchsten Genauigkeit aus und speichert es als neue Basislocation. Die *WeightedAverage*-Klasse filtert ebenfalls zuerst zu alte Standortdaten und führt dann eine auf Basis der Genauigkeit gewichtete Mittelwertbildung der Koordinaten durch und speichert diese neuen Koordinaten als Basiclocation.

[53] http://sourcemaking.com/design_patterns/strategy (Stand: 29.06.2010)

6.5.3 Validierung und Qualitätszustand

Die Details, wie sich mit LCS zu verbinden ist, dieser konfiguriert und gestartet wird, ist nicht Aufgabe des LocationEstimator, sondern wird vollständig vom LCSM übernommen. Der LE übermittelt dem LCSM nur welche Services mit welcher Konfiguration ausgewählt werden sollen, und an welches Callback Status- und Locationupdates gesendet werden sollen.

Trotzdem benötigt der LE Informationen über die verfügbaren Ortungsdienste und ihre Eigenschaften. Außerdem muss der LE nach Eintreffen eines Locationupdates den Qualitätsstatus bestimmen und aktualisieren (Vergleich Seite 5.9). Dazu gibt es die Klasse *LocationServiceInformation* im package *de.tud.qalmw.le*, welche vom LCSM erstellt wird, die Id, Servicebeschreibung und Zustand des entsprechenden LCS enthält und den Qualitätszustand kapselt.

Der LE kann daraus die Eigenschaften entnehmen und bei Eintreffen eines Locationupdates mit der Methode validateService die Qualitätsprüfung durchführen. Hat sich dabei eine Änderung am Zustand ergeben, wird die Methode lcsValidationChanged der SelectionStrategy aufgerufen.

6.6 Anbindung Ortungsdienste: LCS und LCSM

Für das Finden und Anbinden der verschiedenen LCS ist der LocationServiceManager im package *de.tud.qalmw.lcsm* verantwortlich. Er versorgt den LE auf Anfrage mit einer aktuellen Liste verfügbarer LCS (gekapselt in der Klasse LocationServiceInformation) und informiert ihn über Statusänderungen der LCS. Außerdem bindet, aktiviert und konfiguriert er die vom LE ausgewählten LCS.

Der LCSM definiert via AIDL das Remote Interface *ILocationService* welches von konkreten LCSs implementiert werden muss. Es enthält Methoden zum registrieren/deregistrieren von Callbacks (*(un)registerLocationCallback*, *(un)registerStatusCallback*), Abfragen des letzten Standords (*getLastKnownLocation*) und Konfigurieren (*setIntervall*). Zur Vereinfachung stellt der LCSM auch eine abstrakte Basisklasse *ALocationService* zur Verfügung. Sie leitet sich von der Android-Service-Komponente *android.app.Service*[54] ab, damit jeder LCS als langlebige Service-Komponente existieren kann und implementiert bereits einige Standardfunktionen wie das Senden von Locationupdates. Jeder konkrete LCS kann direkt von dieser Klasse ableiten und sein dienstspezifisches Verhalten hinzufügen und eine Ortungslösung kapseln.

Um die lose Kopplung zwischen den LCS und der Middleware zu realisieren wurde auf den Android-Mechanismus der Intents zurückgegriffen. Jeder LCS definiert in seinem Manifest das Intent *de.tud.qalmw.lcsm.ILocationService*. Über eine Androidsystemfunktion (*PackageManager.queryIntentServices*) kann der LCSM eine Liste aller Services abfragen die auf dieses Intent

[54] http://developer.android.com/reference/android/app/Service.html (Stand: 27.06.2010)

Listing 6.1: Definition eines LCS in der zugehörigen AndroidManifest.xml

```
<service android:name=".lcs.GpsLcs"
    android:process=":remote"
    android:label="GPS Location Service">
    <intent-filter>
        <action android:name="de.tud.qalmw.lcsm.ILocationService" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    <intent-filter>
        <action android:name="de.tud.qalmw.lcs.GpsLcs" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    <meta-data android:name="supportsCoordinates" android:value="true" />
    <meta-data android:name="supportsAltitude" android:value="true" />
    <meta-data android:name="referenceSystem" android:value="wgs-84" />
    <meta-data android:name="supportsSymbolic" android:value="false" />
    <meta-data android:name="supportsSpeed" android:value="true" />
    <meta-data android:name="supportsDirection" android:value="true" />
    <meta-data android:name="ttff" android:value="30" />
    <meta-data android:name="accuracy" android:value="10.0" />
    <meta-data android:name="latency" android:value="1000" />
    <meta-data android:name="costs" android:value="3" />
    <meta-data android:name="charges" android:value="0" />
</service>
```

reagieren, und damit eine Liste aller auf dem Gerät verfügbaren und kompatiblen LCS erhalten. Jeder LCS definiert darüber hinaus einen eigenen eindeutigen Intent (typischerweise sein vollständiger Klassenname), über den er vom LCSM schließlich aktiviert und angebunden werden kann. Damit sind LCS und die Middleware getrennt. Es können jederzeit neue LCS implementiert und auf dem Gerät installiert werden. Der LCSM wird diese bei erneuter Suche ebenfalls finden. Das System ist damit flexibel um neue Ortungsdienste erweiterbar.

Jeder LCS stellt seine Dienstbeschreibung zur Verfügung, in dem er in seiner Manifest-Definition (AndroidManifest.xml) die Diensteigenschaften als Metadaten spezifiziert. Der LCSM kann auf diese Metadaten nach Abfragen der Dienstliste zugreifen und so die *ServiceDescription* erstellen, welche im weiteren Verlauf ausgewertet werden kann. Die Dienstbeschreibung steht damit bereits zur Verfügung, auch ohne dass die LCS instanziiert werden müssen, was ressourcenschonend ist. Auflistung 6.1 zeigt ein Beispiel für die Manifest-Definition eines LCS.

Um eine Verbindung zu einem Remote Service herzustellen und dessen Methoden aufrufen zu können, muss die Android-Komponente *ServiceConnection*[55] implementiert werden. Die Klasse *RemoteServiceConnection* im package *de.tud.qalmw.lcsm* ist leitet sich von *ServiceConnection* ab und ist dafür verantwortlich eine LCS anzubinden. Der LCSM erstellt für jeden gefunden LCS ein solches Objekt, und speichert darüber hinaus die *ServiceDescription*. Wird ein Service gewählt

[55] http://developer.android.com/reference/android/content/ServiceConnection.html (Stand: 27.06.2010)

kann über seine ServiceConnection Verbindung aufgebaut werden (bindService) und Methoden ausgeführt werden.

Der LE führt Methoden an LCS nicht direkt aus sondern beauftragt den LCSM. Dazu stehen u. A. die Methoden *selectServices* und *unselectServices* aus, die als Eingabeparameter eine Liste mit ID's der LCS erwartet. Der LCSM sucht sich darauf hin die entsprechenden RemoteServiceConnection-Objekte und führt das Konfigurieren und Registrieren/Deregistrieren aus.

Um Informationen über verfügbare Services zu erhalten kann der LE vom LCSM eine Liste abfragen mit dem geforderten Datentyp als Eingabeparameter. Zurückgegeben wird eine Liste von *LocationServiceInformation*-Objekten, welche die Dienstbeschreibung und vom LE zu ermittelndem Qualitätszustand kapseln (Vergleich Kapitel 5.8)

6.7 Ablaufdetails

Der Gesamtablauf findet um den LE herum statt und entspricht der Darstellung in Abbildung 5.4 aus Kapitel 5.3. Abbildung 6.5 verdeutlicht das Zusammenspiel der einzelnen Komponenten.

Sobald ein neuer LBS-Request hinzugefügt wurde (1a) wird zunächst eine LocationResponse mit der aktuell verfügbaren Basislocation gesendet (1b). Entspricht diese bereits den Anforderungen der LBS-Anfrage und liegt ein Singlerequest vor, wird der Vorgang mit Übermittlung des OperationStatus finished bereits beendet.

Ansonsten wird der Auswahlalgorithmus angestoßen, in dem seine requestAdded-Methode aufgerufen wird. Vom LCSM wird eine Liste verfügbarer LCS abgefragt (2a). Dieser hat auf dem Gerät befindliche LCS entdeckt und ihre ServiceDescription abgerufen (3a) und kann dem LE die entsprechenden Informationen senden (2b). Nach durchlaufen des SelectionAlgorithmus werden ein oder mehrere LCS ausgewählt (2c) und vom LCSM aktiviert (3b). Es sei denn, es sind keine Dienste verfügbar, dann wird die LBS-Anfrage mit einer negativen Locationresponse beendet (1b).

Sobald das erste LocationUpdate eines LCS eintrifft (4), wird es im DataManager gespeichert (5a), der Estimationsalgorithmus durchlaufen, welcher auf die Daten des DM zugreift (5a) und eine LocationResponse gesendet (1b). Parallel zur Estimation wird der Dienst des eingetroffenen Updates validiert. Änderungen am Qualitätszustand werden durch aufruf der Methode lcsValidationChanged vom Selectionsalgorithmus behandelt, was möglicherweise zu einer veränderten Dienstauswahl führt (2c).

Statusänderungen eines LCS werden beim LCSM gemeldet (3d) und an den LE weitergeleitet, welcher durch Aufruf der Methode lcsStatusChanged des Selectionalgorithmus behandelt werden können.

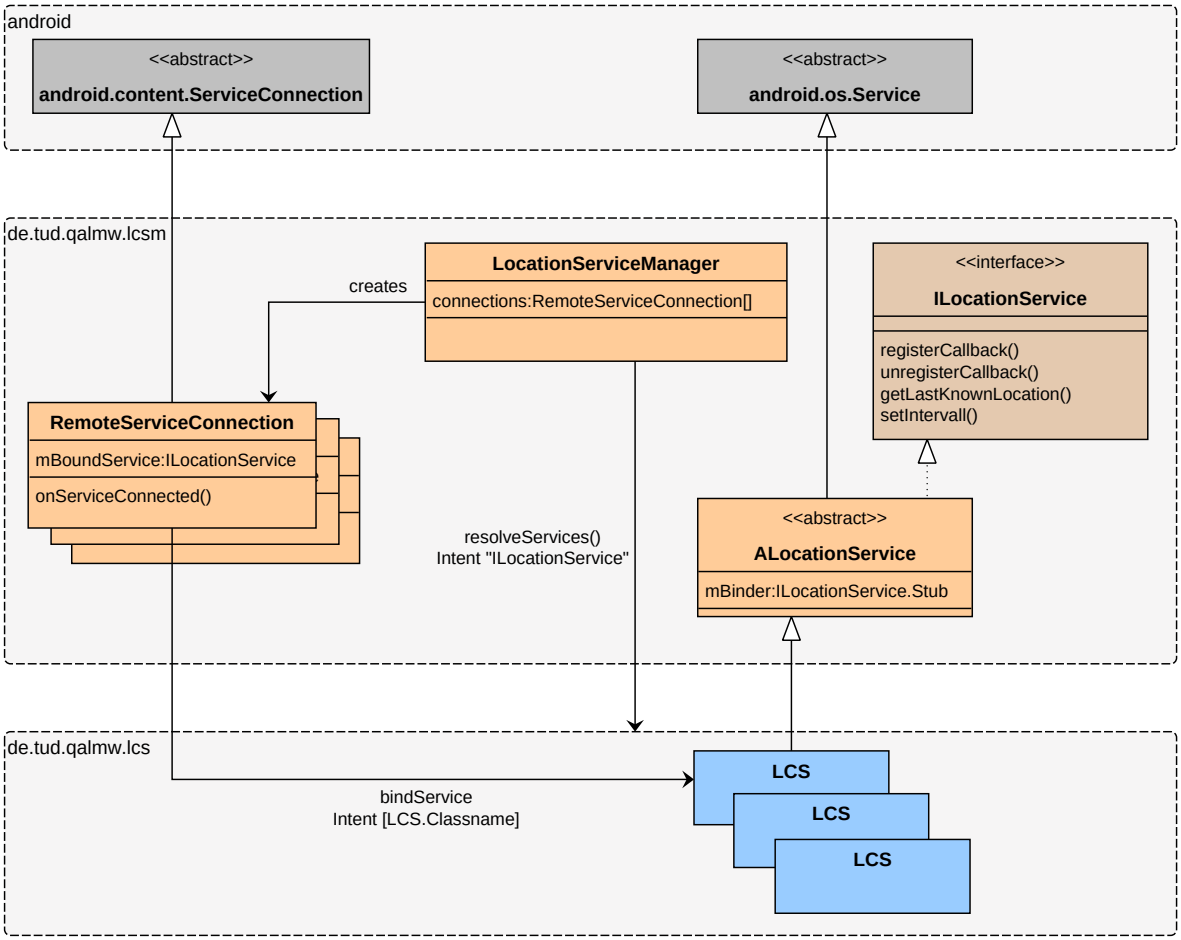


Abb. 6.4: Packageverteilung und wichtige Klassen im Zusammenspiel LCSM und LCS

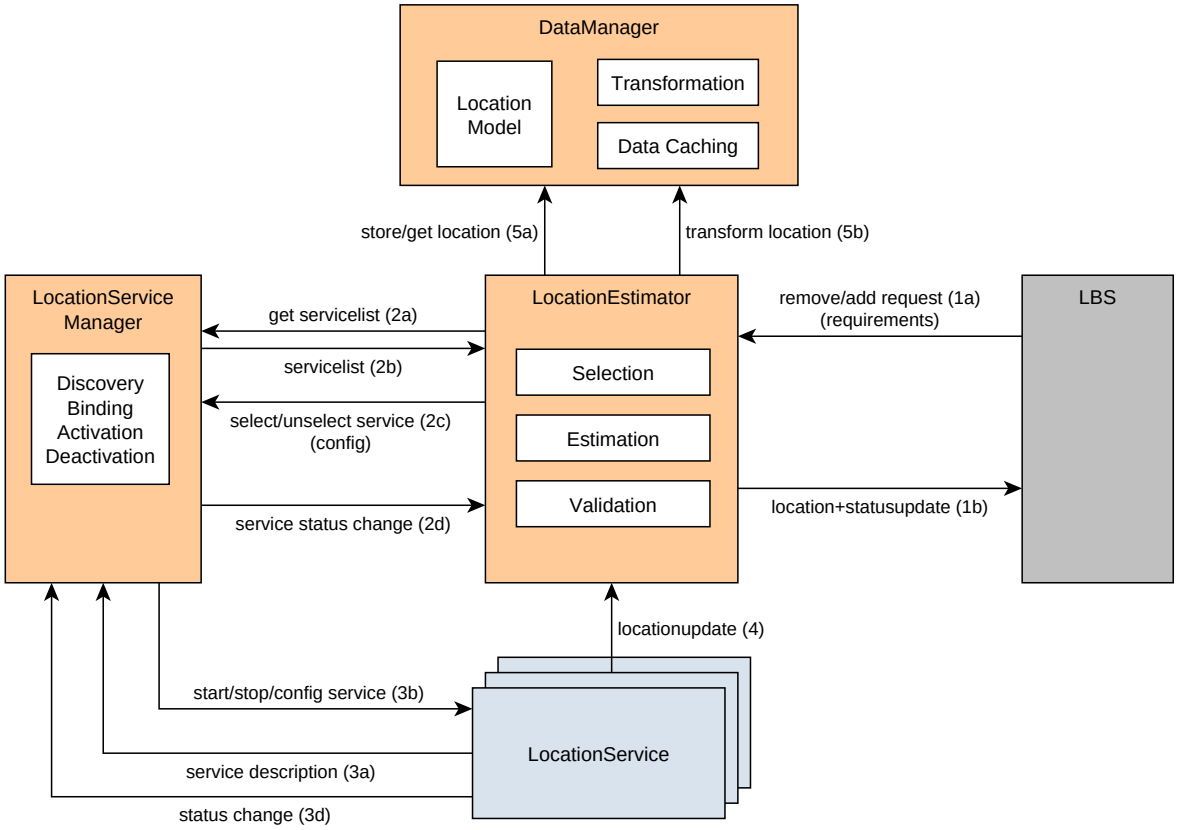


Abb. 6.5: Ablaufdetails

Wird der Request schließlich entfernt (1a), wird die Methode requestRemoved des Selectionsalgorithmus aufgerufen, in deren Ergebnis ausgewählte Dienste beim LCSM abgewählt (2c), und von ihm gestoppt werden (3b).

6.8 Testanwendungen

Um die Funktionalität der Middleware Testen und Untersuchen zu können wurden zwei Testanwendungen, *GetLocationText* und *GetLocationMap* erstellt, welche sich im package *de.tud.qalmw.test*

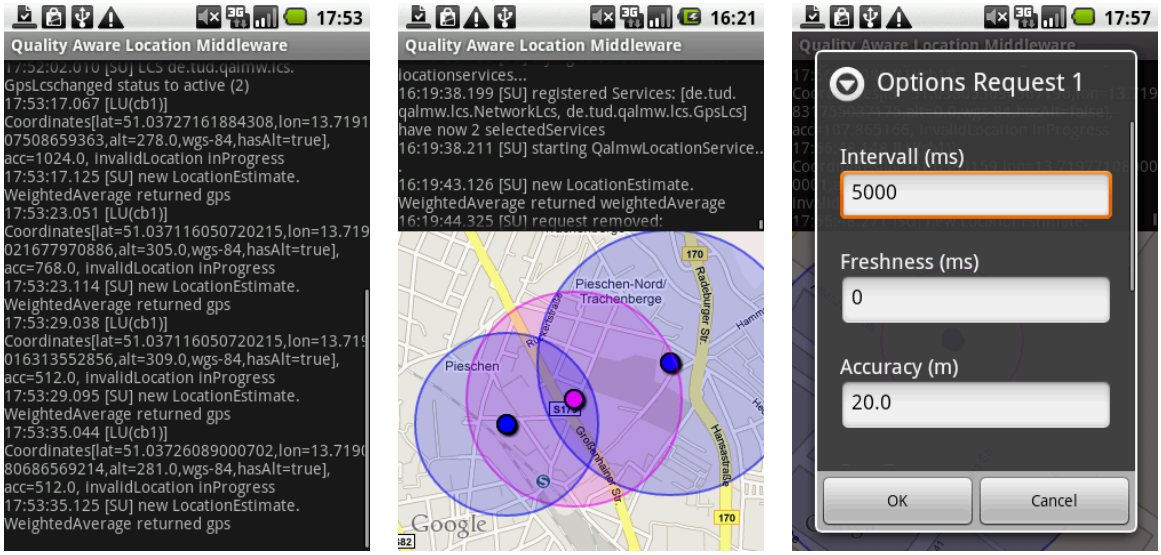


Abb. 6.6: Screenshots der beiden Testanwendungen. Links GetLocationText, mitte GetLocationMap, rechts Optionsdialog

befinden. Beide leiten sich von der Android-Komponente Activity[56] bzw. MapActivity[57] ab. Sie verfügen über ein Textfeld, in dem empfangene Locationupdates und Statusupdates angezeigt werden. GetLocationMap verfügt darüberhinaus über eine Kartenansicht, in welcher die Locationupdates der verschiedenen LCS und der Basiclocation dargestellt werden. Dazu wird die jeweilige Position auf der Karte markiert, und die Genauigkeit als Umkreis visualisiert.

Über das Menü lassen sich zwei Dialoge öffnen, in denen die Anforderungen für zwei verschiedene Request spezifiziert werden können. Zu den einstellbaren Parametern zählen Interval, Accuracy, Freshness, Datatype, Requesttype. Nachdem über eine Menüauswahl explizit eine ServiceConnection zur Middleware hergestellt wurde können die beiden Requests hinzugefügt und wieder entfernt werden.

Über weitere Menüoptionen kann die Estimationsstrategie und die Selectionstrategie gewählt werden. Außerdem lässt sich einstellen ob Locationupdates, Statusupdates oder beides in die Textfelder ausgegeben werden soll. Schließlich kann ein im LE integriertes Loggingsystem aktiviert werden, welches Statusausgaben und eingetroffene LocationUpdates in eine Textfile auf der Speicherkarte des Gerätes speichert.

[56] http://developer.android.com/reference/android/app/Activity.html (Stand: 26.06.2010)
[57] http://code.google.com/intl/de-DE/android/add-ons/google-apis/reference/com/google/android/maps/MapActivity.html (Stand: 26.06.2010)

7 Evaluation

Mit der Implementierung des Prototyps wurde die Voraussetzung geschaffen, das entwickelte Konzept zu überprüfen und zu bewerten. Im Folgenden sollen einige Ergebnisse und Erkenntnisse ausgewertet werden.

7.1 Ortungsdienste

Für den Test des Prototypen wurden als Ortungsdienste die beiden Android Locationprovider GPS und Network verwendet. Damit wurde sowohl satellitenbasierte Ortung als auch Mobilfunk- (Cell-ID) und WLAN-Ortung integriert.

Allerdings bietet die WLAN-Ortung der Google Location Services nur eine niedrige Qualität (niedrige Genauigkeit, keine kleinen echtzeitfähigen Abfrageintervalle), und bleibt weit unter dem Potential für WLAN-Ortung, vor allem für den Innenraumbereich.

Um die die volle Leistungsfähigkeit der Middleware ausreizen und testen zu können ist eine größere Anzahl an Ortungsdiensten mit einem wesentlich breiteren Spektrum an Eigenschaften notwendig.

Zwar verfügt das verwendete HTC Dream über zahlreiche Sensoren (Accelerometer, Kompass, Lagesensor), es konnte allerdings noch keine Lösung gefunden und als LCS angebunden werden.

Da das System aber um neue Ortungsdienste erweiterbar konzipiert und umgesetzt wurde, besteht hier noch großes Potential für weiterführende Untersuchungen und Experimente.

7.1.1 Die LCS des Prototyps

Um aus den verfügbaren Diensten das beste herauszuholen und die Bandbreite an Diensteigenschaften zu vergrößern wurde der Networkprovider in zwei verschiedenen LCS gekapselt. Die Klasse *NetworkCellidLcs* simuliert eine Nur-Cell-ID-Lösung, während die Klasse *NetworkWifiLcs* die Wlan-Ortung simuliert. Um dies zu realisieren, überwachen beide Klassen über den den Android WifiManager[58] und einen BroadcastReceiver[59] den Zustand der WLAN-Verbindung

[58] http://developer.android.com/reference/android/net/wifi/WifiManager.html (Stand: 27.06.2010)
[59] http://developer.android.com/reference/android/content/BroadcastReceiver.html (Stand: 27.06.2010)

des Gerätes und setzen entsprechend ihren Zustand. Das Aktivieren/Deaktivieren der WLAN-Verbindung führt also zum Wechseln der Verfügbarkeit der beiden LCS.
Es handelt sich dabei allerdings um eine Kompromisslösung. Da beide den gleichen Locationprovider nutzen, kann es, obwohl WLAN aktiv ist, dazu kommen, dass der NetworkWifiLcs Locationupdates auf Mobilfunk-Cell-ID-Basis ermittelt und überträgt, wenn keine WLAN-Netzweite in Reichweite sind.
Der GpsProvider wurde in der Klasse *GpsLcs* gekapselt.

7.2 Dienstauswahl

Für den Prototyp wurden zwei verschiedene Strategien zur Dienstauswahl implementiert. Während *SelectAll* alle, für einen Datentyp verfügbaren Dienste aktiviert, und vornehmlich dazu gedacht ist, die EstimationsStrategien zu testen, wurde mit *BalancedSelection* ein Algorithmus implementiert, der den Versuch darstellt, eine Optimalauswahl zu erreichen.

Nachdem über den LCSM eine Liste verfügbarer und das geforderte Format unterstützende Dienste bereitsteht, wird durch einen Algorithmus versucht, die Liste nach verschiedenen Kriterien zu sortieren. Das Ziel dabei ist, dass an oberster Stelle Dienste stehen, welche die Anforderungen des LBS-Requests gerade erfüllen können, jedoch die niedrigsten Kosten und Aufwand verursachen. Für die Umsetzung wurden die Kriterien Genauigkeit, Intervall und Kosten verwendet.

In Tabelle 7.1 sind die drei LCS gelistet mit den jeweiligen relevanten Kriterien. Außerdem sind zwei Beispielrequests mit den entsprechenden Anforderungen gelistet. Die Anforderungen der Requests an die Kosten sind implizit als null gegeben. Zu den Requests ist außerdem angegeben, in welche Reihenfolge der Algorithmus die LCS sortieren sollte. Beispielsweise erfüllen NetworkWifiLcs und GpsLcs die Genauigkeitsanforderungen des ersten Requests, weshalb sie in der Liste vor dem NetworkLcs stehen. Die Intervallanforderungen erfüllen auch beide. Da NetworkWifiLcs aber niedrigere kosten aufweist als GpsLcs, steht er in der Liste an erster Stelle.

Zur Demonstration der Dienstauswahl und der Reaktion des Algorithmus auf Statusänderungen der LCS könnte man sich folgendes Szenario überlegen: Beide Provider sind enabled. Außerdem ist WLAN aktiviert. Entsprechend sind GpsLcs und NetworkWifiLcs im Zustand ready und NetworkCellidLcs im Zustand deactivated. Beim Starten wird zunächst NetworkWifiLcs ausgewählt. Deaktiviert der Nutzer WLAN ändert sichd er Zustand von NetworkWifiLcs auf paused und die Auswahl springt weiter zu GpsLcs. Deaktiviert man GPS springt die Auswahl weiter zu NetworkCellidLcs. Reaktiviert man WLAN springt die Auswahl zurück zu NetworkWifiLcs.
Listing A.1 zeigt einen Ausschnitt aus dem Middleware-Logs und wie sich bei den ganannten Ereignissen der Status der Ortungsdienste ändert und die Dienstauswahl angepasst wird.

	Accuracy	Latency (Interval)	Costs	Balanced List
GpsLcs	5	1000	3	
NetworkCellidLcs	1000	60000	1	
NetworkWifiLcs	100	60000	2	
Request 1	120	60000	(0)	[NetworkWifiLcs, GpsLcs, NetworkCellidLcs]
Request 2	20	5000	(0)	[GpsLcs, NetworkWifiLcs, NetworkCellidLcs]

Tab. 7.1: LCS-Eigenschaften, Beispielrequests und resultierende Sortierung

7.2.1 Verbesserungen.

Der vorgestellte Algorithmus sorgt dafür, dass immer genau ein Dienst, und zwar der Beste, entsprechend der Sortierung gewählt ist. Randbedingungen, zum Beispiel während der Initierungsphase der Standortbestimmung werden noch nicht betrachtet. So könnte es sein, dass an erster Stelle GPS ausgewählt wird, was auf Grund des möglicherweise höheren TTFF zu einer längeren Wartezeit auf das erste Locationupdate führen könnte. Hier könnte man die Liste nach einem weiteren Dienst durchsuchen, der eine kurze TTFF bietet, und als zweite Auswahl aktivieren, damit schnell ein erstes Locationupdate bereit steht. Sobald die Erstauswahl in einem stabilen Qualitätszustand gekommen ist (steady) kann der zweite Dienst deaktiviert werden.

In der Umsetzung wurden die Kriterien fest vorgegeben und in der Reihenfolge Accuracy, Interval, Costs beachtet. Um mehr Flexibilität zu ermöglichen sollte der Algorithmus dahingehend erweitert werden, dass die zu betrachtenden Kriterien (und Reihenfolge) vom anfragenden LBS spezifiziert werden werden können.

7.2.2 Alternative Gesamtranking

Der vorgestellte Algorithmus erstellt die Liste in einem iterativen Verfahren in dem er Teile davon immer weiter eingrenzt und umsortiert. Eine Alternative wäre es, jeden Service anhand der Kriterien und seiner Vorgaben aus der Dienstbeschreibung zu bewerten, so dass er eine

Gesamtnote erhält nach welcher eine Gesamtrangliste erstellt werden kann. Der Request gibt dabei die Kriterien vor und mit welchem Gewicht sie in das Ranking eingehen sollen.

Für jeden Dienst und jedes Kriterium könnte ein Erreichbarkeitswert berechnet werden, d. h. ein vergleichbarer Zahlenwert, der aussagt ob und vor allem wie hoch die Anforderung des Requests erfüllt wird. Hat man Beispielsweise drei Dienste mit Genauigkeit 20m, 100m, 1000m, und die Anforderung lautet 50m, dann leistet der erste Dienst eine Erreichbarkeit von mehr als 100%, während die anderen beiden Dienste darunter liegen. Die Frage stellt sich, wie man die Wertebereiche festlegt, um die anteilige Erreichbarkeit der einzelnen Dienste zu bestimmen.

Kann man dieses mathematische Problem lösen, lassen sich die Erreichbarkeitswerte und Gewichte recht einfach zusammenfassen und ein Gesamtranking erstellen, durch welches die Auswahl wie im vorgestellten Ansatz ausgeführt werden kann.

7.2.3 Zusammenhang Genauigkeit - Intervall

Eine interessante Frage stellt sich nach dem Zusammenhang und der Beeinflussung von Genauigkeit und Intervall. Beispielsweise macht bei einer möglichen, niedrigen Genauigkeit von 1000 Meter ein kleines Abfrageintervall von einer Sekunde keinen Sinn, da die Wahrscheinlichkeit, in einer Sekunde eine Entfernung von 1000 Metern zurückgelegt zu haben, sehr niedrig ist (wenn man die Bewegung zu Fuß oder maximal mit dem Auto als Grundlage annimmt). Andererseits, bei großem Abfrageintervall von beispielsweise einer Minute bringt eine hohe Genauigkeit von wenigen Metern kaum einen Mehrwert, denn die feingranularen Ortsveränderungen werden nur sehr grob erfasst.

7.3 Aggregation

In Kapitel 5.3.3 wurde vorgestellt, wie Standortdaten mehrerer, gleichzeitig aktivierter Ortungsdienste aggregiert werden können. Dabei wurden die beiden grundlegenden Ansätze Auswahl und Fusion unterschieden. Bei Auswahl (SelectBestAccuracy) wird dasjenige Locationupdate ausgewählt, was innerhalb eines gewissen Zeitlimits ermittelt wurde und die höchste Genauigkeit aufweist. Bei Fusion werden die verschiedenen Locationupdates verschmolzen, indem der gewichtete Mittelwert der Einzelkoordinaten gebildet wird (WeightedAverage).

7.3.1 Verschmelzung

Im Test zeigte sich, dass, wie bereits vermutet wurde, die einfache gewichtete Mittelwertbildung auf Basis der Genauigkeit allein nicht ausreicht, um zu verhindern, dass qualitativ hochwertige Updates durch niederwertige Updates verschlechtert werden. Abbildung 7.1a und b zeigen die Ergebnisse

a) b)

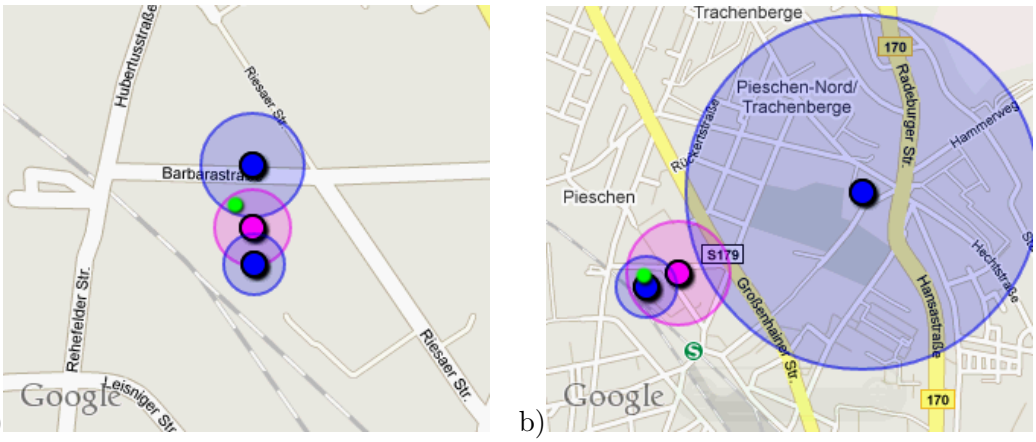


Abb. 7.1: Verschmelzung von Locationupdates mit ähnlicher Genauigkeit (links) und unterschiedlicher Genauigkeit (rechts)

(magenta) der Verschmelzung zweier Locationupdates (blau) mit den Genauigkeitsumkreisen und die tatsächliche Position (grün). Haben beide Locationupdates in etwa die gleiche Qualität (a), bringt die neu berechnete Position einen Informationsvorteil. Unterscheidet sich die Qualität jedoch erheblich, zum Beispiel auf Grund des sehr großen Genauigkeitsunterschieds wie in (b), weicht die neu berechnete Position von der tatsächlichen Position weiter ab, als das Bessere der beiden Einzelupdates. Es kommt zu einer Verschlechterung des Endergebnisses.
Es scheint also sinnvoll zu sein, nur in bestimmten Fällen eine Verschmelzung durchzuführen. Die Untersuchung der Ähnlichkeit der Genauigkeit allein reicht dazu allerdings nicht aus.

Eine weitere Beobachtung liefert interessante Erkenntnisse über das Verhalten und die Zuverlässigkeit der verschiedenen Ortungsdienste. In Abbildung 7.2 sind wieder je zwei Locationupdates, das fusionierte Ergebnis und die tatsächliche Position abgebildet. Das jeweils linke Update wurde vom GPS ermittelt, weil sich das Gerät innerhalb eines Gebäudes befand allerdings versehen mit einem niedrigen Genauigkeitswert (großer Umkreis). Das rechte Update wurde durch Mobilfunkortung (Cell-ID) ermittelt.

In beiden Bildern erkennt man, dass GPS beide Male zwar eine sehr schlechte Genauigkeitsinformation übermittelt, prinzipiell mit der tatsächlichen Position aber relativ genau übereinstimmt. Dagegen ist das Ergebnis der Mobilfunkortung beide Male komplett falsch. Die Genauigkeit allein sagt also noch nichts über die prinzipielle Zuverlässigkeit eines Ortungsdienstes und seiner ermittelten Standortdaten.

In beiden Fällen sieht man, dass die Verschmelzung zu einer enormen Verfälschung führen kann und daher abzulehnen ist. Stattdessen sollte dasjenige Locationupdate ausgewählt werden, dessen Ortungsdienst eine höhere Zuverlässigkeit hat, und daher am wahrscheinlichsten ist, zuzutreffen.

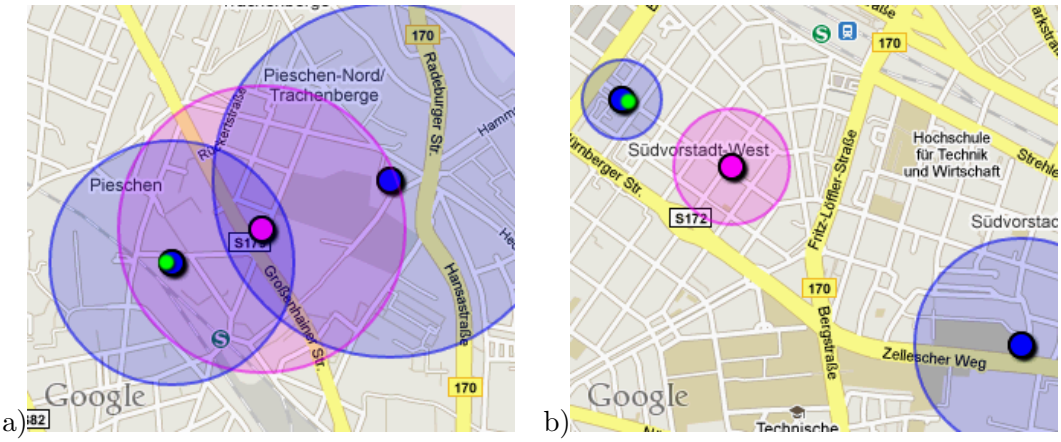


Abb. 7.2: Verschmelzung von Locationupdates mit ähnlicher Genauigkeit aber großem Abstand

Ein verbesserter Aggregationsalgorithmus sollte also Auswahl und Fusion dynamisch kombinieren, indem zunächst eine Auswahl getroffen und unzuverlässige Dienste ausgesondert werden, und erst die verbleibenden Updates verschmolzen werden.

7.3.2 Testlauf

Es wurde ein Testlauf durch die Altstadt von Dresden durchgeführt. Der Weg führte vom Hauptbahnhof (In der Darstellung unten) die Prager Straße entlang, einmal unter der Prager Zeile hindurch (Hochhauskomplex, zu erkennen am Schlenker nach rechts in der Mitte der Strecke), zurück zur Prager Straße, und in die Centrum Gallerie (in der Darstellung oben, Grundriss hervorgehoben.) und wieder hinaus.
Als Auswahlstrategie wurde SelectAll verwendet, damit alle verfügbaren LCS (Adroids GPS und Networkprovider, mit WLAN falls verfügbar) genutzt werden. Als Aggregationsalgorithmus wurde der WeightedAverage genutzt. Abbildung 7.3 zeigt die aufgezeichneten Pfade vom Networkprovider (links), Gpsprovider (mitte) und ermittelten Basiclocation (rechts). Das Abfrageintervall wurde auf fünf Sekunden gesetzt.

Die WLAN-Abdeckung war geringer als erhofft. Die höchste Genauigkeit wurde gelegentlich mit um die 75m erreicht (Genauigkeit ist farblich gekennzeichnet, in der linken Abbildung ist die Skala dargestellt). Auf Grund des vorgegebenen hohen Abfrageintervalls von bestenfalls einer Minute, kann diese Lösung kaum Verbesserungen beitragen, falls GPS in der Qualität nachlässt. Man sieht im linken Bild wie die ermittelten Positionen oft hin- und herspringen, zwischen gelegentlich verfügbaren WLAN Access Points und Mobilfunkstationen.

In den Außenbereichen hat sich deshalb GPS fast vollkommen durchgesetzt, da es sehr hohe

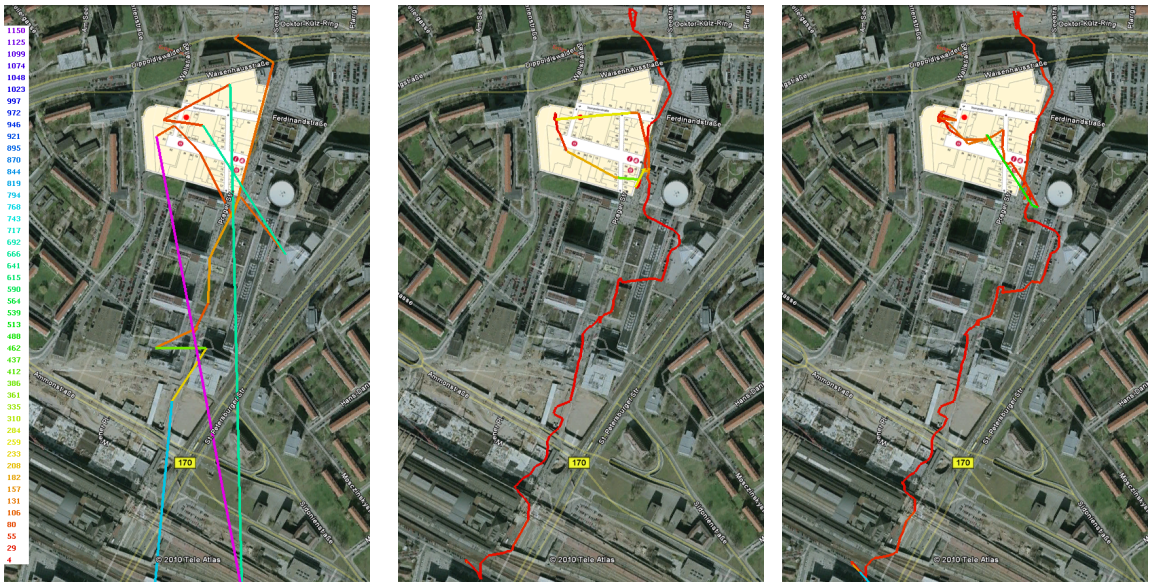


Abb. 7.3: Testlauf Dresden Innenstadt, links: Network, mitte: GPS, rechts aggregierte Basiclocation

Genauigkeit lieferte und entsprechend hoch gewichtet in die ermittelte Basiclocation eingegangen ist.

Innerhalb der Centrum Galerie war nur sehr eingeschränkter GPS-Empfang möglich, weshalb der Networkprovider mehr Einfluss auf das Endergebnis hatte. Die Basiclocation entspricht dort eher dem tatsächlich gelaufenen Pfad, als das reine GPS. Hier ist eine kleine Verbesserung zu erkennen. Dennoch bleibt das Ergebnis auf Grund der genannten Probleme des benutzten Networkprovider unter den Möglichkeiten der Middleware. In einem solchen Szenario würde eine lokal vorhandene, qualitativ hochwertige Innenraumortung (hohe Genauigkeit, kurze Intervalle), welche über die Middleware dynamisch eingebunden werden kann, das GPS ersetzen und eine deutliche Verbesserung bringen.

7.4 Synchronisation

Bei einer Anfrage für kontinuierliche Standortupdates gibt der LBS ein Intervall vor, in welchem er regelmäßig Locationupdates empfangen will. Nicht immer liefert ein LCS Locationupdates exakt im geforderten Intervall. Sind sogar mehrere Dienste gewählt, welche in unterschiedlichen Intervallen und Zeitpunkten Locationupdates senden, kann sich das negativ auf das gleichmäßige und kontinuierliche Senden der LocationResponses an LBS auswirken.

Hier muss abgewägt werden, ob das vom LBS geforderte Intervall auf jeden Fall, so gut wie möglich eingehalten werden soll, oder nur als Richtwert für die Ansteuerung darunterliegender LCS benutzt wird.

Würde man die Benachrichtigung eines LBS über einen Timer steuern, würden dieser zwar regelmäßige Updates erhalten, es könnten aber veraltete Standortdaten gesendet werden. Im schlimmsten Fall wurde gerade ein LBS-Update gesendet, welches ein veraltetes Locationupdate enthält, und kurz darauf trifft von einem LCS ein neues Update ein, welches wiederum erst nach Ablauf der Intervallphase gesendet wird.
Würde man stattdessen den LBS immer umgehend benachrichtigen sobald ein neues Update eines LCS eingetroffen ist, führt das zwar zu unregelmäßigeren Updates, welche dafür aber immer hochaktuell sind.
Die Dynamik der LCS und fehlender Garantien über Latenzen lässt kaum zuverlässige Abschätzungen machen, um dieses Problem zu umgehen.

7.5 Herausforderungen der Implementierung

Für die Umsetzung des Prototyps wurde sich intensiv mit der Entwicklungsplattform Google Android beschäftigt. Dabei mussten auch einige Probleme und Unzulänglichkeiten umgegangen werden, die im Folgenden erwähnt werden sollen.

7.5.1 Benachrichtigungen über Statusänderungen der Ortungsdienste

Damit jeder Ortungsdienst seinen Zustand aktuell halten kann, und die Middleware umgehend über Änderungen informieren kann, muss er jederzeit wissen, ob noch alle Bedingungen für seine Nutzbarkeit gegeben sind und selbst über Änderungen dynamisch informiert werden. Für die Android Locationprovider bedeutet das unter anderem, zu überwachen, ob sie vom Nutzer über die Systemeinstellungen aktiviert/deaktiviert wurden. Diese Information kann explizit über den LocationManager abgefragt werden.

Asynchrone Benachrichtigungen darüber werden allerdings nur gesendet, solang ein implementierter LocationListener beim LocationManager registriert ist, was im Fall eines LCS nur gemacht wird, wenn der Dienst ausgewählt und aktiviert wurde. Dynamische Benachrichtigungen über die Verfügbarkeit sind also nur für den Fall möglich, dass ein Dienst deaktiviert wurde.

Um festzustellen, ob ein deaktivierter (und nicht ausgewählter) Dienst wieder reaktiviert wurde, bleibt nur die Möglichkeit eines Pollings. In der Umsetzung wurde dies implementiert, d.h. solang der jeweilige LCS nicht ausgewählt und aktiviert ist, fragt er in regelmäßigem Abstand beim LocationManager ab, ob er aktiviert wurde. Das Intervall ist 10 Sekunden voreingestellt. Wird der Dienst ausgewählt, wird das Polling wieder deaktiviert.

7.5.2 ServiceConnection und Callbackmechanismus

Um mit einem (Remote) Service zu kommunzieren und Funktionen aufrufen zu können, muss vom Client eine *ServiceConnection* implementiert werden. Mit Übergabe des entsprechenden Intents an die Systemfunktion *bindService* wird der Verbindungsaufbau initiiert. Hat Android schließlich eine Verbindung hergestellt wird die Methode *onServiceConnected* des Client aufgerufen und das ServiceStub-Objekt[60] übergeben. Der Gegenpart zu onServiceConnected heißt *onService-Disconnected*. Diese Methode wird allerdings nicht wie erwartet aufgerufen, wenn der Client über unbindService die Verbindung trennen will, sondern nur wenn die Verbindung unerwartet abbricht, weil der Service zum Beispiel abgestürzt ist. Man muss also bei manueller Trennung der Verbindung selbst sicherstellen, dass das ServiceStub-Objekt nichtmehr nutzbar ist.

Damit ein Remoteservice asynchrone Nachrichten an seine Clients schicken kann (z. B.: Locationupdates) muss zuvor ein Callback-Objekt registriert werden. Zur Verwaltung von *RemoteCallbacks* bietet Android eine eigene Klasse, die *RemoteCallbackList*[61], welche u. A. überwacht, wenn ein Zielprozess verschwindet, und threadsicheren Zugriff auf die Liste gewährt. Unglücklicherweise gibt es keine Benachrichtigung, wenn ein Zielprozess regulär beendet wird, sondern ebenfalls nur, wenn er abstürzt. Es muss daher darauf geachtet werden, dass vor Beenden des Zielprozesses das RemoteCallback manuell wieder entfernt wird.

Um Aufrufe an den registrierten Callbacks durchführen zu können gibt es die Methode *begin-Broadcast*. Damit es zu keinen Nebenläufigkeitsproblemen kommt, darf nur ein solcher Broadcast gleichzeitig durchgeführt werden. Auf Grund der vielen Statusänderungen im Auswahlprozess kann ein solcher Fall schnell eintreten. Der Aufruf sollte daher über einen Handler[62] gesteuert werden, welcher über einen Messagequeue verfügt und damit die Benachrichtigungsaufgaben nacheinander durchführen kann.

[60] Details zum RPC-Mechanismus (Remote Procedure Call) in Android unter http://developer.android.com/guide/developing/tools/aidl.html (Stand: 28.06.2010)
[61] http://developer.android.com/reference/android/os/RemoteCallbackList.html (Stand: 28.06.2010)
[62] http://developer.android.com/reference/android/os/Handler.html (Stand: 28.06.2010)

8 Zusammenfassung und Ausblick

Im ersten Teil dieser Arbeit wurden die Grundlagen und Charakteristiken von LBS und der zu ihrer Ausführung notwendigen automatisierten Bestimmung des Standorts herausgearbeitet.

Anschließend wurde mit der Anforderungsanalyse für eine Ortungsmiddleware detaillierter auf die Problemstellung eingegangen. Es wurden Anwendungsszenarien vorgestellt und grundlegende Probleme identifiziert und erklärt, die bei Suche, Auswahl, Ausführung und Kombination verschiedener Ortungsdienste auftreten können. Außerdem wurde analysiert, welche wissenschaftlichen, kommerziellen und standardisierten Lösungsansätze es bereits gibt, und welche Schwächen oder Potentiale sie aufweisen.

Mit den gesammelten Erkenntnissen und Anforderungen wurde schließlich das Konzept für eine Ortungsmiddleware entwickelt und vorgestellt. Der Fokus lag dabei auf der Entwicklung einer flexiblen und erweiterbaren Gesamtarchitektur. Es wurde detailliert auf die einzelnen Komponenten und ihr Zusammenspiel unter verschiedenen Aspekten eingegangen. Für die wichtigen Kernkomponenten Dienstauswahl (Selection) und Standortbestimmung (Estimation) wurden einige Beispielansätze vorgestellt.

Anschließend wurde auf Basis der Android Plattform ein lauffähiger Prototyp implementiert. Durch einen Test auf dem HTC Dream konnten die Kernkonzepte gezeigt werden. Dennoch wurde mit dieser Arbeit erst eine Grundlage geschaffen für eine Vielzahl weiterer Forschungen und Experimente.

Es wurde zwei Ortungsdienste verwendet und integriert: die auf dem HTC Dream verfügbaren Locationprovider Gps und Network des Android Location Frameworks. Sie konnten dynamisch von der Middleware gefunden und angebunden werden.
Für die Demonstration der Flexibilität und Wirkungsweise der Kernalgorithmen stellt dies aber nur eine kleinen Vorgeschmack dar. Es sollten daher weitere Ortungsdienste integriert und einbezogen werden um mit einem breiteren Spektrum an Diensteigenschaften Experimente durchführen zu können. Vor allem sollte eine hochwertige WLAN-Ortungslösung verwendet werden um der Übergang vom Außen- zum Innenbereich wirkungsvoll demonstrieren zu können.
Ebenfalls sollte das Potential der verfügbaren Sensoren durch Einbindung in konkrete Ortungslösungen erforscht werden.
Auch der in Kapitel 5.1 erwähnte RemoteLocationServiceManager bedarf einiger Forschung. Viele Implementierungen sind denkbar, welche Ortungslösungen lokaler Netzwerke in die Middleware integrieren - Flexibel und dynamisch zur Laufzeit.

Bei den Kernkomponenten wurden verschiedene Auswahlalgorithmen und Aggregationsmechanismen implementiert und gestestet. Es wurde die Funktionalität gezeigt erste Beobachtungen über die Wirksamkeit gemacht. Auch hier gibt es noch großes Erweiterungspotential.
Intelligentere Aggregationsalgorithmen, die verschiedene Datentypen und Formattransformationen einbeziehen, zum Beispiel von einer Sensorlösung ermittelte Bewegungsdaten, sollten integriert werden. Dabei sollten sowohl Auswahl als auch Datenfusion kombiniert werden, je nach vorliegenden Daten, und weitere Parameter wie allgemeine Zuverlässigkeit und zeitliche Dämpfung einbezogen werden. Aber auch gänzlich andere Aggregationsmechanismen könnten in die Middleware integriert und durch Experimente verglichen werden
Die Dienstauswahl kann ebenfalls noch flexibler gestaltet werden, indem weitere Randbedingungen einbezogen werden.

Die Grundvoraussetzungen für weitere Forschung und Experimente sind mit der entwickelten Architektur gelegt.

Eine wichtige Anforderung wurde zu Beginn des Konzepts zurückgelegt: das Sammeln und Vereinen der unterschiedlichen Anforderungen mehrerer LBS. Die sogenannte Requestaggregation steigert nochmals die Komplexität einer solchen Middleware und sollte in weiteren Arbeiten detaillierter untersucht werden.

Zusammenfassend lässt sich feststellen, dass die meisten Anforderungen erfüllt werden konnten. Es wurde ein flexibles erweiterbares System konzipiert, für ein echtes Smartphone umgesetzt und mit verschiedenen Beispielansätzen getestet.
Leider war die Menge der verfügbaren Ortungsdienste recht überschaubar, weshalb die Fähigkeiten der Middleware nicht ausgereizt werden konnten.

Mit neuen, vielfältigen Ortungsdiensten sowie erweiterten und alternativen Estimations- und Auswahlstrategien kann die Middleware ausgebaut werden und bietet noch viel Potential für weiterführende Forschung und Experimente.

A Anhang

Kategorie	**Beschreibung**	**Beispiele**	**push** oder **pull**	**self** oder **cross**
Informationen	Nutzer erfragt Informationen bezogen auf seinen Standort	Lokale Wettervorhersage, Navigation, lokale Karten, ÖPNV Abfahrtszeiten	Pull	self
Point of interest	Der Nutzer sucht nachegelegene stationäre Objekte, Einrichtungen	Restaurant- , Hotel-, Geschäfts-Finder, Dienstleistungssuche (Drucker...)	Pull	self
Auffinden anderer Nutzer	Der Nutzer sucht in der Nähe befindliche andere Nutzer	Spiele, Freundefinder, Flirt-finder, "Buddyfinder"	Pull	self & cross
Tracking	Ein (eher stationärer) Nutzer überwacht Standort anderer mobiler Personen/Objekte	Flottenmanagement, Überwachung von Kindern, Patienten, Ärzten, Objekten (Assets)	Pull (überwachender Nutzer)	cross
Notfalldienste	Service-center empfängt Standort eines mobilen Anrufers welcher Hilfe benötigt	Notruf, Pannendienst	Push (Notfalldienst) Pull (Notrufer)	cross (Notfalldienst)
Messaging & Announcement	Mobile Nutzer empfangen Nachrichten von anderen Nutzern die in einem bestimmten Gebiet senden	Lokale Werbung, Nachrichten an benachbarte Freunde	Push	self & cross
Auslöser	Ein mobiler Nutzer empfängt einen Auslöser wenn er einen bestimmten Bereich betritt	Standortbezogene Erinnerungen, Verkehrswarnung, Wetterwarnung	Push	self
Gebühren-erfassung	Ein Nutzer kriegt Rechnung entsprechend seinem Standort	Gebührenabrechnung, Homezone	Push	self

Tab. A.1: Kategorien von LBS (nach [Rot05, SGG08])

property	values
functional capabilities - location format	
supportsCoordinates	true or false
supportsAltitude	true or false
supportsSymbolic	true or false
ontology	String
supportsSpeed	true or false
supportsDirection	true or false
functional capabilities - request type	
supportsSingle	true or false
supportsPeriodic	true or false
supportsProactive	true or false
nonfunctional capabilities - static quality properties	
latency	value as millisecond
charges	no, low, med, high
costs	no, low, med, high
nonfunctional capabilities - dynamicc quality properties	
ttff	value as millisecond
accuracy	value as meter

Tab. A.2: Dienstbeschreibung mit LCS-Parametern

Listing A.1: Ereignislog der Middleware zur Dienstauswahl und Statusänderungen der LCS

```
1  LCSM found 3 services: [de.tud.qalmw.lcs.NetworkCellidLcs, de.tud.qalmw.lcs.GpsLcs, de.
       tud.qalmw.lcs.NetworkWifiLcs]
2  new balancedList(3) for req[acc=120.0,interval=60000,costs=lowest]=[
3      de.tud.qalmw.lcs.NetworkWifiLcs(acc=100.0,interval=60000,costs=2),
4      de.tud.qalmw.lcs.GpsLcs(acc=10.0,interval=1000,costs=3),
5      de.tud.qalmw.lcs.NetworkCellidLcs(acc=1000.0,interval=60000,costs=1),]
6  selected Services: [de.tud.qalmw.lcs.NetworkWifiLcs]
7      have now 1 selectedServices: [de.tud.qalmw.lcs.NetworkWifiLcs] index=0
8  LCS de.tud.qalmw.lcs.NetworkWifiLcs changed status to active (2)
9
10 // user disables wlan
11 LCS de.tud.qalmw.lcs.NetworkCellidLcs changed status to ready (1)
12 LCS de.tud.qalmw.lcs.NetworkWifiLcs changed status to paused (3)
13 selected Services: [de.tud.qalmw.lcs.GpsLcs]
14     have now 2 selectedServices: [de.tud.qalmw.lcs.GpsLcs, de.tud.qalmw.lcs.
           NetworkWifiLcs] index=1
15 unselected Services: [de.tud.qalmw.lcs.NetworkWifiLcs]
16     have now 1 selectedServices: [de.tud.qalmw.lcs.GpsLcs] index=1
17 LCS de.tud.qalmw.lcs.NetworkWifiLcs changed status to deactivated (0)
18 LCS de.tud.qalmw.lcs.GpsLcs changed status to active (2)
19
20 // user disables gps
21 LCS de.tud.qalmw.lcs.GpsLcs changed status to paused (3)
22 selected Services: [de.tud.qalmw.lcs.NetworkCellidLcs]
23     have now 2 selectedServices: [de.tud.qalmw.lcs.NetworkCellidLcs, de.tud.qalmw.lcs.
           GpsLcs] index=2
24 unselected Services: [de.tud.qalmw.lcs.GpsLcs]
25     have now 1 selectedServices: [de.tud.qalmw.lcs.NetworkCellidLcs] index=2
26 LCS de.tud.qalmw.lcs.GpsLcs changed status to deactivated (0)
27 LCS de.tud.qalmw.lcs.NetworkCellidLcs changed status to active (2)
28
29 // user enables wlan
30 LCS de.tud.qalmw.lcs.NetworkCellidLcs changed status to paused (3)
31 unselected Services: [de.tud.qalmw.lcs.NetworkCellidLcs] have now 0 selectedServices: []
       index=2
32 LCS de.tud.qalmw.lcs.NetworkCellidLcs changed status to deactivated (0)
33 LCS de.tud.qalmw.lcs.NetworkWifiLcs changed status to ready (1)
34 selected Services: [de.tud.qalmw.lcs.NetworkWifiLcs]
35     have now 1 selectedServices: [de.tud.qalmw.lcs.NetworkWifiLcs] index=0
36 unselected Services: [de.tud.qalmw.lcs.NetworkCellidLcs]
37     have now 1 selectedServices: [de.tud.qalmw.lcs.NetworkWifiLcs] index=0
38 LCS de.tud.qalmw.lcs.NetworkWifiLcs changed status to active (2)
```

Literaturverzeichnis

[BCG08] Paolo Bellavista, Antonio Corradi, and Carlo Giannelli. The PoSIM middleware for translucent and context-aware integrated management of heterogeneous positioning systems. *Comput. Commun.*, 31(6):1078–1090, 2008.

[BL09] Allan Brimicombe and Chao Li. *Location-Based Services and Geo-Information Engineering*. Wiley & Sons, 1., auflage edition, June 2009.

[BP09] Arno Becker and Marcus Pant. *Android: Grundlagen und Programmierung*. dpunkt Verlag, 1 edition, May 2009.

[Bur09] Ed Burnette. *Hello, Android: Introducing Google's Mobile Development Platform*. Pragmatic Programmers, 0002 edition, November 2009.

[DG07] Severin Olloz Dominik Gruntz. Erfahrungen mit dem location api (jsr 179). Technical report, Institut für Mobile und Verteilte Systeme Fachhochschule Nordwestschweiz, 2007.

[FBDH08] Renato Filjar, Lidija Busic, Sasa Desic, and Darko Huljenic. LBS position estimation by adaptive selection of positioning sensors based on requested QoS. In *Proceedings of the 8th international conference, NEW2AN and 1st Russian Conference on Smart Spaces, ruSMART on Next Generation Teletraffic and Wired/Wireless Advanced Networking*, pages 101–109, St. Petersburg, Russia, 2008. Springer-Verlag.

[Hor07] Matthias Horbank. Adaptive ortsbezogene dienste. Master's thesis, Humboldt-Universität zu Berlin Institut für Informatik Lehrstuhl für Rechnerorganisation und Kommunikation, Juni 2007.

[Hus09a] Thomas Husson. The future of location-based services: Four service categories will emerge. Technical report, Forrester, June 2009.

[Hus09b] Thomas Husson. Western european mobile forecast, 2009 to 2014. Technical report, Forrester, August 2009.

[HWJT09] Rene Hansen, Rico Wind, Christian S. Jensen, and Bent Thomsen. Seamless Indoor/Outdoor positioning handover for Location-Based services in streamspin. In *Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 267–272. IEEE Computer Society, 2009.

[IH05] Peter Ibach and Matthias Horbank. Highly available Location-Based services in mobile environments. In *Service Availability*, pages 134–147. 2005.

[iSu07] iSuppli. Shipments of gps-enabled mobile handsets to more than quadruple by 2011. Technical report, iSuppli Corporation, 2007.

[iSu09] iSuppli. One-third of mobile phones to use accelerometers by 2010, spurred by iphone and palm pre. Technical report, iSuppli Corporation, 2009.

[JSH07] Rainer Joeckel, Manfred Stober, and Wolfgang Huep. *Elektronische Entfernungs- und Richtungsmessung und ihre Integration in aktuelle Positionierungsverfahren*. Wichmann, 5., neu bearbeitete und erweiterte auflage. edition, December 2007.

[KJ06] Krzysztof Kolodziej and Hjelm Johan. *Local Positioning Systems: LBS applications and services*. CRC Press Inc, illustrated edition edition, May 2006.

[Küp05] Axel Küpper. *Location-based Services : Fundamentals and Operation*. John Wiley & Sons Ltd., 1., auflage edition, October 2005.

[KTL06] A. Küpper, G. Treu, and C. Linnhoff-Popien. TraX: a device-centric middleware framework for location-based services. *Communications Magazine, IEEE*, 44(9):114–120, 2006.

[Mur09] Mark L. Murphy. *Beginning Android*. Apress, new. edition, June 2009.

[PB06] Carlo Giannelli Paolo Bellavista, Antonio Corradi. Enhancing jsr-179 for positioning system integration and management. In *First Workshop on Distributed Agent-based Retrieval Tools (DART'06)*, Pula-Cagliari, Sardinia, Italy, 2006.

[RAC+04] Anand Ranganathan, Jalal Al-Muhtadi, Shiva Chetan, Roy Campbell, and M. Dennis Mickunas. MiddleWhere: a middleware for location awareness in ubiquitous computing applications. In *Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 397–416, Toronto, Canada, 2004. Springer-Verlag New York, Inc.

[Rot05] Jörg Roth. *A Decentralized Location Service Providing Semantic Locations*. PhD thesis, Fernuniversität Hagen, 2005.

[SGG08] Zafer Sahinoglu, Sinan Gezici, and Ismail Guvenc. *Ultra-wideband Positioning Systems: Theoretical Limits, Ranging Algorithms, and Protocols*. Cambridge University Press, illustrated edition edition, September 2008.

[SSE06] Moritz Neun Stefan Steiniger and Alistair Edwardes. Foundations of location based services (lecture notes). Department of Geography, University of Zurich, Switzerland, 2006.

[wgs00] Department of defense world geodetic system 1984: Its definition and relationships with local geodetic systems. third edition. Technical report, National Imagery and Mapping Agency, 2000.

Abbildungsverzeichnis

Tabellenverzeichnis

Auflistungsverzeichnis

Abkürzungsverzeichnis

Ajax	Asynchronous JavaScript and XML
API	Application Programming Interface
CSS	Cascading Style Sheets
DM	Data Manager
GIS	Geoinformationssystem
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTML	Hypertext Markup Language
IMSI	International Mobile Subscriber Identity
LBS	Location-based Service
LCS	Location Service
LCSM	Location Service Manager
LE	Location Estimator
MSISDN	Mobile Subscriber Integrated Services Digital Network Number
PDA	Personal Digital Assistent
POI	Point of Interest
QoS	Quality of Service
UMTS	Universal Mobile Telecommunications System
WAP	Wireless Access Point
WLAN	Wireless Local Area Network