Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektbericht

Peter Salchow

Pervasive Gaming Framework

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Peter Salchow

Pervasive Gaming Framework

Ausarbeitung im Rahmen des Projekts
im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuer: Prof. Dr. Olaf Zukunft

Abgegeben am 6. März 2009

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung

Die Vorliegende Ausarbeitung berichtet über das Projekt „Pervasive Gaming“, welches im Wintersemester 2008/2009 im Masterstudiengang Informatik an der HAW Hamburg durchgeführt wurde.

1.1 Motivation

Die zunehmende Verbreitung mobiler Endgeräte, wie z. B. PDAs und Mobiltelefone, führte dazu, dass die Leistungsfähigkeit dieser Geräte stark zunahm. Zusätzlich wurden sie mit Technologien ausgestattet, die weit über die reinen Funktionen des Mobilfunks hinausgehen. Dazu zählen nicht nur GPS, Bluetooth, WLAN und Kompass, sondern auch die Öffnung von Schnittstellen zur Anwendungsentwicklung. Mit diesen Voraussetzungen ist es möglich, das mobile Endgerät, durch die Auswertung ortsbezogener Daten, in die reale Umwelt zu integrieren. Darüber hinaus können diese Informationen mit anderen Geräten ausgetauscht werden. Durch diese Gegebenheiten bekam auch der Bereich der pervasiven Spiele neuen Vortrieb. Komplexe Anwendungen konnten jetzt auf mobilen Endgeräten ausgeführt werden. Dabei wird die reale Umwelt mit virtuellen Elementen verknüpft. Die im Spiel ausgeführten Tätigkeiten werden durch sämtliche Einflüsse aus der realen Umwelt beeinflusst [vgl. Dedaj u. Preisler, 2009].

Im Rahmen des Masterprojekts sollte daher ein Framework zur Erstellung von pervasiven Spielen auf mobilen Endgeräten entwickelt werden. Der Spielbereich diente dabei als Metapher für die Erstellung pervasiver Anwendungen. Dadurch können viele Erfahrungen gesammelt und der Umgang mit Problemen auf diesem Gebiet erlernt werden.

1.2 Aufbau der Arbeit

Nach der Beschreibung des Aufbaus und des Ablaufs des Projekts in Kapitel 2, wird in Kapitel 3 die Entwicklung des Frameworks näher beschrieben. Dabei werden zunächst die Anforderungen an das Framework erläutert. Anschließend wird der Aufbau und die Architektur vorgestellt. Dem Autor war es dabei wichtig, die einzelnen Schritte des Entwurfs aufzuzeigen.

Am Ende des Kapitels dann auf den Aufbau des Kommunikationsmoduls näher eingegangen. In Kapitel 4 wird ein Fazit gezogen und ein Ausblick für weitere Arbeiten gegeben.

2 Das Projekt

Im Masterstudiengang Informatik an der HAW Hamburg, wird im dritten Semester ein Projekt angeboten, in dem die Studierenden in eigener Verantwortung ein Thema bearbeiten sollen. Der Umfang des Projekts beträgt acht Semesterwochenstunden. In dieser Zeit sollen Konzepte und Lösungen zu einem Thema erarbeitet werden. Im folgenden wird über das Projekt „Pervasive Gaming“ berichtet.

2.1 Projektaufbau

Das Projekt wurde von Dennis Dedaj, Amine El-Ayadi, Arazm Hosieny, Tobias Hutzler, Sascha Kluth, Julia Plischka, Thomas Preisler und Peter Salchow durchgeführt. Betreut wurde es durch Prof. Dr. Olaf Zukunft.

Bereits in den ersten Projektmeetings wurde beschlossen, dass ein Framework zur Erstellung von pervasiven Anwendungen entwickelt werden soll. Zur Bestätigung der Funktionsfähigkeit sollten zusätzlich zwei Anwendungen entwickelt werden, die auf dem Framework aufbauen. Aufgrund dieser Beschlüsse entwickelte sich auch die Einteilung der Teams.

Dennis Dedaj, Sascha Kluth und Thomas Preisler übernahmen den Entwurf und die Entwicklung des Rollenspiels ‘"The World Within’", auf Basis des Frameworks. Eine genaue Beschreibung der Ergebnisse kann in den Berichten Dedaj u. Preisler [2009] und Kluth [2009]nachgelesen werden. Amine El-Ayadi und Julia Plischka entschieden sich dafür eine kollaborative Kalenderanwendung für Mobiltelefone zu entwickeln, mit der wichtige Termine und Ereignisse in einem Netzwerk ausgetauscht werden können. Dieses Teilprojekt lief unter dem Namen ‘ïLife’". Die Berichte Plischka [2009] und El-Ayadi [2009] geben hierzu weitere Informationen. Arazm Hosieny, Tobias Hutzler und Peter Salchow übernahmen die Entwicklung des Frameworks, auf dem die Anwendungen aufbauen. Sie hatten das Ziel, eine lauffähige Umgebung zu erschaffen, die eine komfortable Erstellung von pervasiven, mobilen Anwendungen ermöglicht. Eine detailliertere Beschreibung des Frameworks erfolgt in den folgenden Kapiteln.

2.2 Projektablauf

Zunächst soll noch kurz über den Ablauf des Projekts berichtet werden. Es fand im Wintersemester 2008/2009 statt und hatte einen Umfang von acht Semesterwochenstunden. In den wöchentlichen Meetings wurden zunächst die Projektziele definiert und die zu verwendenden Technologien ausgewählt. Später wurden dann einzelne Teilaufgaben erstellt und mit einem Fertigstellungszeitpunkt an einen Bearbeiter im Team vergeben. Teilergebnisse und neue Erkenntnisse wurden ebenfalls in den Meetings präsentiert. Alle Meetings wurden in Protokollen dokumentiert.

2.3 Technologien

Zu Beginn des Projekts wurde die zu verwendende Plattform und das dazu gehörende Endgerät bestimmt. Zur Auswahl standen Windows Mobile, das iPhone und Android. Das iPhone wurde von den Projektmitgliedern abgelehnt, da einige die Einarbeitung in die unbekannte Programmiersprache Objective C als zu großes Risiko betrachteten. Probleme mit der Programmiersprache würden von der wesentlichen, konzeptionellen Arbeit ablenken. Die Motivation der Gruppe, Windows Mobile zu benutzen, war sehr gering. Auch hier wurden Bedenken gegenüber der Programmiersprache C# geäußert. Aus diesem Grund wurde die Verwendung von Android beschlossen. Bei dieser Plattform war die Motivation der Gruppe am größten. Alle wollten die noch sehr junge Plattform kennenlernen. Hinzu kam, dass Android die favorisierte Programmiersprache Java verwendet und ein open-source Projekt ist. Der einzige Nachteil von Android war, dass der Auslieferungszeitpunkt der Endgeräte noch nicht fest stand. Es war fraglich, ob die Anwendungen noch in der Projektzeit auf dem realen Endgerät getestet werden können. Dennoch hat sich das Projektteam für Android entschieden und eine mögliche Ausführung der Anwendungen im Emulator in Kauf genommen.

Android ist ein Software-Stack für mobile Geräte, der aus einem Betriebssystem, einer Middelware und den darauf laufenden Anwendungen besteht. Mit Android wird eine Umgebung zum Ausführen von Anwendungen auf mobilen Geräten zur Verfügung gestellt. Diese Umgebung abstrahiert von der darunter liegenden Hardware und stellt den Anwendungen Schnittstellen für den Zugriff auf das Gerät zur Verfügung. Alle Anwendungen werden in Java geschrieben [vgl. Google].

3 Framework

In diesem Kapitel wird der Entwurf und die Entwicklung des Frameworks für pervasive mobile Anwendungen näher beschrieben. Dazu werden zunächst die an das Framework gestellten Anforderungen beschrieben. Anschließend wird der Aufbau des Frameworks näher erläutert.

3.1 Anforderungen

Die Anforderungen an das Framework ergaben sich zum größten Teil aus den einzelnen Anforderungen der Anwendungen. Die Gemeinsamkeiten der Anforderungen wurden zusammengefasst und ergaben somit die Anforderungen an das Framework.

3.1.1 Anforderungen durch das Rollenspiel

Im Rollenspiel müssen die Charaktere (Avatare) der Spieler gespeichert und überall zugreifbar sein. Diese fachliche Anforderung beinhaltet zwei technische Anforderungen an das Framework. Zum Einen ist das die Persistierung von Daten (in diesem Fall der Avatare). Zum Anderen wird durch die Anforderung, dass die Avatare überall zugreifbar sein sollen, ein zentraler Server benötigt. Dies wiederum erfordert einen Datenaustausch zwischen Endgerät und Server.

Eine weitere Anforderung durch das Rollenspiel war es, einen Duellkampf zwischen zwei Spielern durchführen zu können. Dazu soll ein Spieler, auf dem Endgerät, einen anderen Spieler auswählen und diesen dann angreifen können. Der angegriffene Mitspieler wird über den Angriff benachrichtigt. Abhängig von Erfahrung und Stärke der Charaktere, wird ein Schaden zugefügt. Um einen Duellkampf durch das Framework zu ermöglichen, muss eine Kommunikation zwischen den Endgeräten stattfinden. Da ein Angriff eine sehr spezielle Funktion des Rollenspiels ist, muss davon soweit abstrahiert werden, dass alle möglichen Daten kommuniziert werden können.

Damit der Duellkampf stattfinden kann, müssen sich die Spieler zur gleichen Zeit in einer bestimmten Entfernung zueinander befinden. Die Avatare werden dazu auf die reale Weltkarte

abgebildet. Dafür ist eine Positionsbestimmung des Endgerätes erforderlich. Das Framework muss die Möglichkeit bieten, die eigene Position und die Position der anderen Mitspieler zu ermitteln. Außerdem müssen alle Mitspieler in einem bestimmten Umkreis ermittelt werden können.

Durch externe Geräte (Wii-Controller) sollen bestimmte Bewegungen und Gesten des Spielers erkannt werden. Diese Bewegungen sollen dann bestimmte Aktionen im Spiel auslösen. Dafür muss das Framework die Daten der Hardware erfassen und auswerten können. Die Anwendung muss dem Framework mitteilen, welche Aktionen einzelnen Bewegungen zugeordnet sind. Das Framework muss anhand des Kontextes und der Bewegungsdaten erkennen, welche Aktion ausgelöst werden soll.

Die oben genannten Anforderungen durch das Rollenspiel sind nur die für das Framework relevanten Anforderungen. Alle Anforderungen an das Rollenspiel können im Bericht Dedaj u. Preisler [2009] nachgelesen werden.

3.1.2 Anforderungen durch iLife

In iLife werden sämtliche Daten (z. B. Bilder, Texte, Ereignisse, Videos) erfasst und im Internet (z. B. in Online-Kalendern und Blogs) veröffentlicht. Die Anwendungen im Internet bieten Schnittstellen zur Bedienung an. Das Framework muss diese Schnittstellen implementieren und die erfassten Daten im Internet veröffentlichen. Da dies eine sehr anwendungsspezifische Anforderung ist, wurde von einer Implementierung im Framework abgesehen. Vielmehr sollen diese Funktionen in Modulen gekapselt werden, die vom Framework sinnvoll verwaltet werden.

Die auf dem Endgerät erstellten Daten (z. B. Bilder, Videos, Texte) sollen mit Geodaten annotiert werden können, damit sowohl zeitliche als auch räumliche Abfolgen rekonstruiert werden können. Das Framework muss dazu die Position des Endgeräts bestimmen können. Da eine Positionsbestimmung innerhalb von Räumen über GPS nicht möglich ist, muss auf andere Verfahren zurückgegriffen werden. Kann die Position auch so nicht bestimmt werden, müssen diese Daten nachträglich bearbeitet werden können.

Die Anwendung iLife ermöglicht es, anderen Benutzern aktuelle Informationen und Ereignisse mitzuteilen. Ein Benutzer kann ein bestimmtes Ereignis in seinen Kalender eintragen. Andere Benutzer, die dies interessiert, werden darüber informiert. Ist ein Benutzer gerade nicht erreichbar, wird er bei der nächsten Möglichkeit benachrichtigt. Das Framework muss dabei Informationen zwischen mehreren Endgeräten austauschen können. Außerdem müssen die Daten (z. B. Ereignisse) an einer zentralen Stelle persistiert werden, damit sie verzögert zugestellt werden können. Die Zuordnung von Ereignissen zu bestimmten Interessen und

andere semantische Auswertungen werden nicht durch das Framework realisiert. Es wurde beschlossen, dass diese Funktionen durch die Anwendung zur Verfügung gestellt werden.

3.1.3 Umzusetzende Anforderungen

Zusammenfassend haben sich für das Framework folgende Anforderungen ergeben:

Persistenz: Beliebige Daten müssen sowohl auf dem Endgerät, als auch auf einem zentralen Server persistiert werden können. Um eine Persistierung auf Android vornehmen zu können, müssen die Objektstrukturen auf die zur Verfügung stehende Datenbank (SQLite) abgebildet werden. Dies erfordert die Implementierung eines OR-Mappers.

Kommunikation: Es müssen beliebige Objekte zwischen den Endgeräten und zwischen Endgerät und Server ausgetauscht werden können. Dabei liegt es in der Verantwortung des Frameworks, den Kommunikationskanal zum richtigen Endgerät aufzubauen. Objekte können nur zwischen gleichen Anwendungen ausgetauscht werden. Das Framework isoliert die einzelnen Anwendungen voneinander.

Positionsdaten: Über das Framework müssen die einzelnen Anwendungen die Möglichkeit haben, die eigenen Positionsdaten und die Positionsdaten anderer Teilnehmer zu ermitteln. Es muss auch möglich sein, Teilnehmer in einem bestimmten Umkreis abfragen zu können. Auch hier muss das Framework darauf achten, dass die Anwendungen bei allen Abfragen getrennt voneinander betrachtet werden.

Bewegungserkennung: Mit Hilfe des Frameworks soll es möglich sein, Bewegungsabläufe des Geräts (oder einer externen Hardware) zu erkennen und diese auszuwerten. Bestimmten Bewegungen können Aktionen zugeordnet werden, die dann vom Framework ausgeführt werden.

Erweiterbarkeit: Das Framework soll modular aufgebaut sein und kann dadurch im Funktionsumfang erweitert werden. Die Verwaltung und Bereitstellung der Daten liegt in der Verantwortung des Frameworks.

3.2 Aufbau des Frameworks

Während des Projektverlaufs haben sich für die Architektur des Frameworks mehrere Entwürfe ergeben. Da bereits von Android ein mächtiges Framework für die Anwendungsentwicklung zur Verfügung gestellt wird, musste ein weiteres Framework entworfen werden, dass auf Android aufsetzt. Die von Android bereitgestellten Funktionen sollten sinnvoll gekapselt und für die Anwendungen erweitert und spezialisiert werden. Abbildung 3.1 zeigt den

ersten Grobentwurf des Framework-Architektur. Dieser Entwurf verdeutlicht die einzelnen

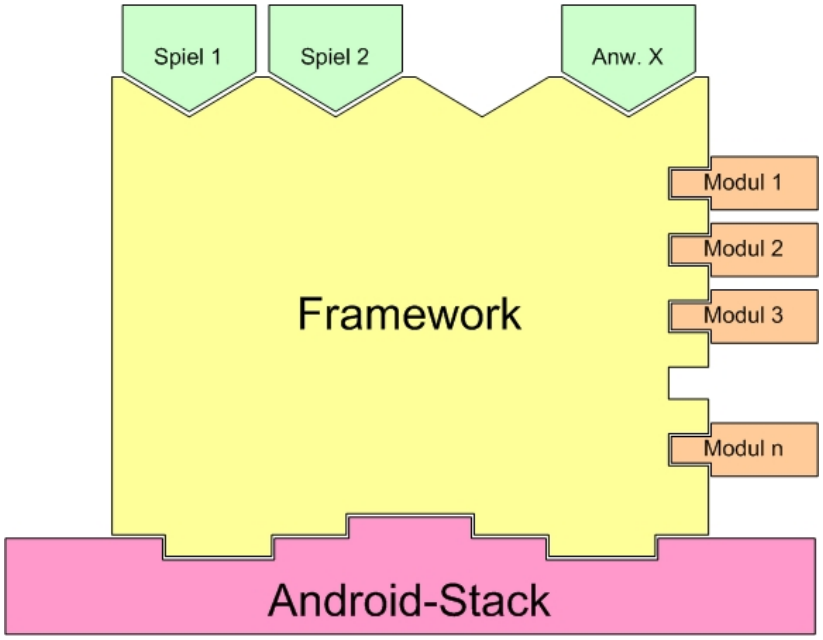

Abbildung 3.1: Grobentwurf des Frameworks

Bestandteile des Frameworks. Zum Einen sind das die Anwendungen die auf dem Framework aufsetzen, zum Anderen sind es Module, die Funktionen bereitstellen. Das Framework selbst hat die Aufgabe, die Module und Anwendungen zu verwalten, die Kommunikation zu steuern und den Zugriff zu überwachen. Mit diesem Aufbau kann ein sehr generelles Framework entwickelt werden. Dies war erforderlich, da von den einzelnen Anwendungen anfangs sehr unterschiedliche Anforderungen erhoben wurden.

In einer weiteren Verfeinerung der Architektur (siehe Abbildung 3.2) wurde der Aufbau des Frameworks näher spezifiziert. Außerdem wurden einige Funktionen bereits näher beschrieben. In diesem Entwurf sollte das Framework aus einer technischen Schicht bestehen, welche die Schnittstelle für die Erweiterungen (Module) bereit stellt. Das Framework selbst besitzt keine fachlichen Funktionen. Die wichtigsten Bestandteile in diesem Entwurf sind das Eventmanagement, der Dependency-Manager und das Modulmanagement. Das Eventmanagement überwacht die Ereignisse, die beispielsweise von den Modulen ausgelöst werden und ruft entsprechende Funktionen in den gerade ausgeführten Anwendungen aus. Der Dependency-Manager überwacht die Abhängigkeiten zwischen den einzelnen Modulen und den Anwendungen. Es wird sichergestellt, dass die von einer Anwendung benötigten Funktionen vom Framework zur Verfügung gestellt werden. Das Modulmanagement übernimmt das Laden der Module und steuert deren Lebenszyklus.

Obwohl die oben beschriebene Architektur zunächst sehr vielversprechend war, wurde sie

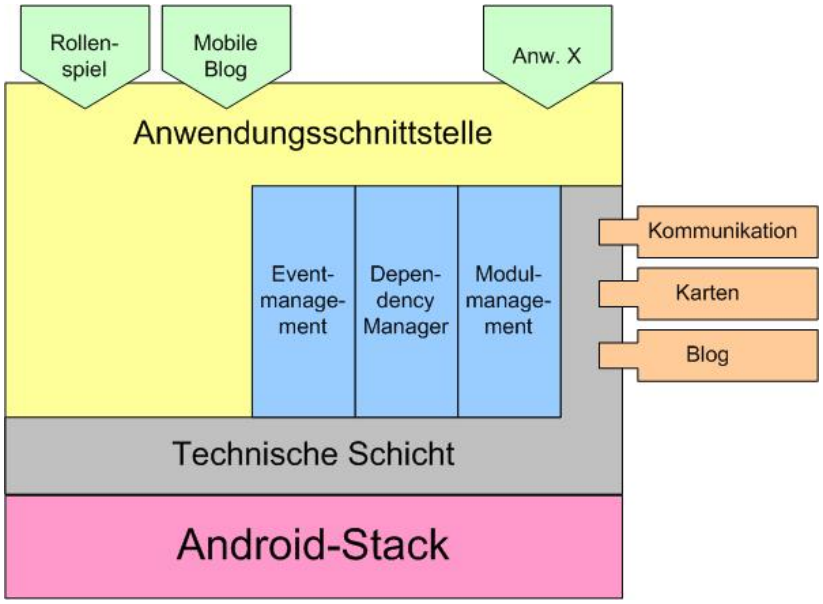

Abbildung 3.2: Verfeinerte Architektur

so nicht umgesetzt. Ein Grund war dafür war der zu hohe Arbeitsaufwand. Die zur Verfügung stehende Zeit hätte nach ersten Schätzungen nicht ausgereicht. Da das Framework die Grundlage für die Arbeit der anderen Teams darstellt, hätten sie erst sehr spät mit ersten Implementierungen beginnen können. Ein anderer Grund war der Mehraufwand durch eine eigne Lifecycle-Kontrolle. In dem oben beschriebenen Ansatz hätte diese im Framework implementiert werden müssen. Da sie aber bereits von Android zur Verfügung gestellt wird, war es sicherer und weniger aufwändig diese Funktionalität von Android zu verwenden.

Im weiteren Projektverlauf, wurde von einem Framework im engeren Sinne abgesehen. Nach dem letztendlich umgesetzten Konzept, werden die Anwendungen als Android-Applikationen entwickelt. Sie können auf die Funktionen des Frameworks über Android-Funktionen zugreifen. Das Framework ist eine Kombination aus speziellen Android-Services. Es stellt darüber die von den Anwendungen geforderten Funktionen zur Verfügung. Die Umsetzung der Funktionen als Services ermöglicht eine dynamische Erweiterung des Funktionsumfangs. Außerdem wird nur eine Instanz von jedem Service benötigt, auf die mehrere Anwendungen gleichzeitig zugreifen können. Android übernimmt das Starten und Beenden der Services. Dadurch werden nur die wirklich benötigten Services gestartet.

3.3 Kommunikationsmodul

Bei der Implementierung des Frameworks wurden zunächst nur die wichtigsten Funktionen umgesetzt. Dazu gehören die Persistenz, die Kommunikation und die Positionierung. Im folgenden Abschnitt wird die Umsetzung des Kommunikationsmoduls näher beschrieben.

3.3.1 Genereller Aufbau

Die Kommunikation zwischen den Anwendungen des Systems findet im jeden Fall über den zentralen Server statt. Möchte ein Client einen anderen kontaktieren, so muss er die Nachricht an den Server schicken, der diese dann verarbeitet und weiterleitet. Ein direkter Datenaustausch zwischen zwei Clients ist nicht möglich. Daher besteht das Kommunikationsmodul aus zwei großen Komponenten; der Serverkomponente und der Clientkomponente. Abbildung 3.3 verdeutlicht diesen Aufbau. Die Clientkomponente setzt auf dem Android-

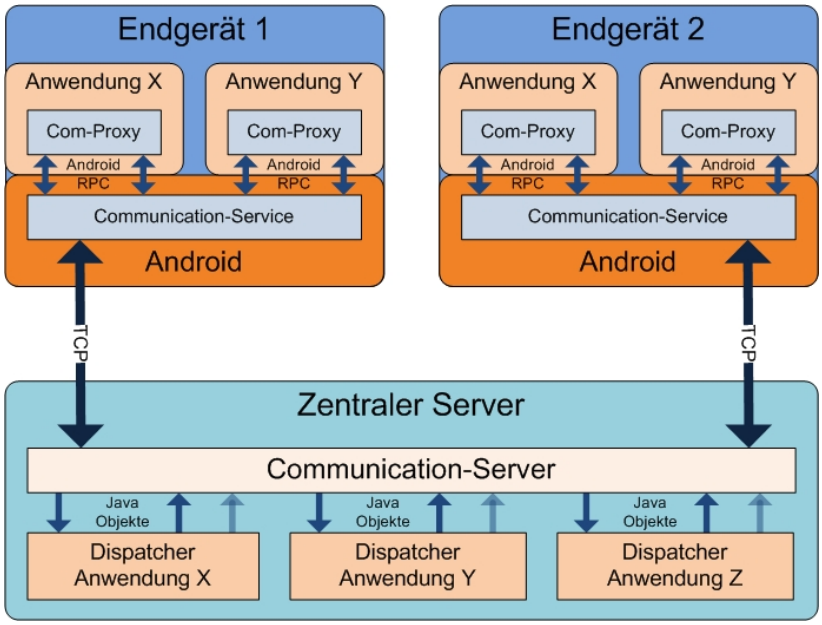

Abbildung 3.3: Architektur des Kommunikationsmoduls

Framework auf und besteht wiederum aus zwei Teilen. Der Hauptteil ist der Communication-Service. Auf jedem Endgerät wird maximal eine Instanz dieses Android-Services gestartet. Für die einzelnen Anwendungen wird ein Com-Proxy zur Verfügung gestellt. Von diesem existiert in jeder Anwendung genau eine Instanz. Der Proxy hat die Aufgabe, die Kommunikation mit dem Service für den Entwickler zu vereinfachen. Er registriert sich beim Service und nimmt Kommunikationsobjekte der Anwendung entgegen. Diese Objekte werden mittels, der

in Android verfügbaren, *Remote Procedure Calls* (RPC) an den Service weitergeleitet. Der Service verarbeitet dieses Objekt und sendet es zum zentralen Server. Die Kommunikation zwischen Server und Service erfolgt über TCP/IP. Ankommende Nachrichten (vom Server) werden vom Service entgegengenommen und dem entsprechenden Proxy zugestellt. Der Proxy übergibt anschließend das Objekt an die Anwendung. Der Service identifiziert den Proxy über eine eindeutige Id.

An der Kommunikationskomponente des zentralen Servers laufen alle Kanäle zusammen. Sie verwaltet alle aktiven Clients und registriert die darauf laufenden Anwendungen. Jede Anwendung registriert auf dem Server einen eigenen Dispatch, der die Verarbeitung der Nachrichten übernimmt. Trifft eine Nachricht von einem Client ein, so wird der zur Anwendung gehörende Dispatcher geladen und die Nachricht an ihn übergeben. Der Dispatcher ist in der Lage, Nachrichten an alle registrierten Clients zu versenden. Dabei werden die einzelnen Dispatcher gegeneinander isoliert. Befindet sich jeder Dispatcher in einem Anwendungskontext, in dem er nur mit gleichen Anwendungen kommunizieren kann. Der Einsatz von Dispatchern ermöglicht eine anwendungsspezifische Anpassung des Kommunikationsmoduls.

3.3.2 Ablauf der Kommunikation

Wie bereits in Abschnitt 3.3.1 beschrieben wurde, besteht zwischen den Clients keine direkte Verbindung, sondern wird über den Server hergestellt. Auf welche Weise bestimmt Objekte ausgetauscht werden verdeutlicht Abbildung 3.4. Wenn eine Anwendung, die das Kommunikationsmodul verwendet, auf dem Endgerät gestartet wird, sendet sie ein Login zu Server und registriert sich auf diese Weise. Das Login wird mit einer Erfolgsmeldung (Login-Ack) bestätigt. Nachdem der Client das Login-Ack erhalten hat, ist der Kommunikationskanal vollständig aufgebaut und kann verwendet werden. Die Anwendung kann dem Kommunikationsmodul jetzt Objekte zum Versand übergeben. Diese werden in Transportobjekte (Datenobjekte) verpackt und zum Server gesendet. Der Server entpackt das Datenobjekt und übergibt es dem entsprechenden Dispatcher. Jede Nachricht, die vom Client zum Server gesendet wird, wird mit einer Nachricht beantwortet. In dieser Nachricht ist ein Antwortobjekt und eine Statusmeldung enthalten. Das Antwortobjekt und der Staus wird vom Dispatcher erzeugt. Das Versenden der Antwort übernimmt das Kommunikationsmodul. Die Beantwortung von Nachrichten erfolgt asynchron. Der Client blockiert nach dem Versand einer Nachricht nicht.

Wenn der Dispatcher aufgerufen wird, kann er Nachrichten (Objekte) an alle Clients der Anwendung schicken. Auch dieser Versand erfolgt asynchron. Nachrichten die vom Server zum Client gesendet werden, werden nicht beantwortet. Für die Adressierung steht dem

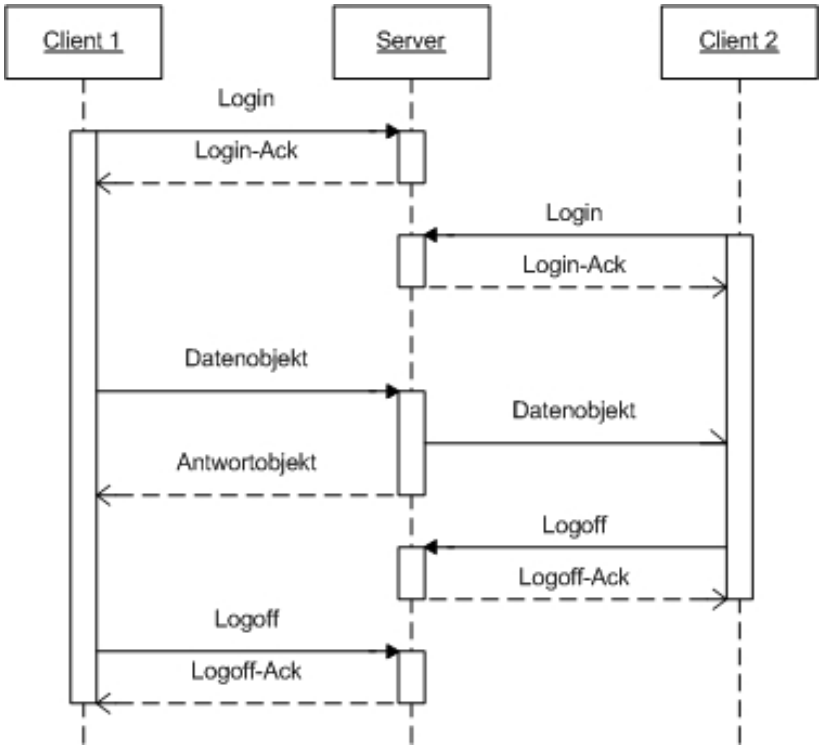

Abbildung 3.4: Ablauf der Kommunikation

Dispatcher eine Liste mit den Namen aller Clients zur Verfügung. Die Zuordnung der Namen zu den entsprechenden Kommunikationskanälen übernimmt das Modul.

Beim Beenden der Anwendung wird, ähnlich wie beim Login, ein Logoff zum Server gesendet. Dieser löscht die Daten des Clients aus seinem Index und sendet als Antwort eine Erfolgsmeldung (Logoff-Ack) an den Client zurück. Der Kommunikationskanal wurde abgebaut. Fällt ein Client aus, bevor das Logoff gesendet wurde, so registriert der Server dies bei einem fehlgeschlagenem Nachrichtenversand oder nach einem Timeout. Auch in diesem Fall wird der Client aus dem Index des Servers entfernt.

Bei der oben beschriebenen Art der Kommunikation ist der Server ein passives Mitglied. Jeglicher Nachrichtenaustausch wird von einem Client initiiert. Es entsteht ein Kommunikationszyklus, der mit dem Versand einer Nachricht durch den Client beginnt und mit der entsprechenden Rückantwort endet. Der Server kann nur in einem solchen Zyklus Nachrichten an andere Clients senden.

4 Fazit und Ausblick

In diesem Kapitel soll ein des Projekts gezogen werden. Abschließend wird ein Ausblick gegeben, wie weiter an dem Projekt gearbeitet werden könnte.

4.1 Fazit

Das Projekt bot jedem Mitglied die Möglichkeit, eigene Interessen und Ideen mit einzubringen. Diese persönlichen Schwerpunkte konnten bei der Umsetzung dann auch bearbeitet werden. Darüber hinaus wurde die Arbeit im Team geschult. Alle Mitglieder waren darauf bedacht, mögliche Risiken zu erkennen und zu umgehen. Die Terminplanung wurde im Team besprochen und von jedem Mitglied weitestgehend eingehalten. Die wöchentlichen Meetings haben sehr positiv dazu beigetragen. Dennoch kam es im Projekt zu Verzögerungen, da einige Risiken falsch bewertet wurden. Auch die Zeiten für die Entwicklung wurden teilweise nicht ganz richtig eingeschätzt. Einige Verzögerungen sind aber auch entstanden, da das G-Phone nicht wie geplant ausgeliefert wurde. Hinzu kamen außerdem noch einige technische Probleme mit der Hardware.

Dennoch konnten zum Ende des Projekts die Ergebnisse erfolgreich präsentiert werden. Der erfolgreiche Einsatz des Rollenspiels hat gezeigt, dass das Framework die Funktionalitäten erfüllt. Die Anwendung „iLife“ konnte auch mit Erfolg präsentiert werden. Allerdings wurde diese Anwendung unabhängig vom Framework implementiert. Die Gründe hierfür waren zeitliche Verzögerungen und Probleme bei Absprachen. Der Projekterfolg wird dadurch aber nicht geschmälert, da eine nachträgliche Migration der Anwendung auf das Framework ohne Probleme möglich ist.

4.2 Ausblick

Die Arbeit in diesem Projekt hat gezeigt, welche Probleme auftreten können und welche Vorgehensweisen geeignet sind, um ein Framework für pervasive Anwendungen zu entwickeln. Einige entworfene Konzepte mussten aufgegeben werden, da andere geeigneter waren. Dadurch konnten im Projekt viele Erfahrungen für ein gutes Konzept gesammelt werden. Am

Ende ist eine solide Basis für weitere Arbeiten entstanden. Das Framework und die beiden Anwendungen sind lauffähig und können erweitert werden.

Einige Projektmitglieder haben bereits Interesse gezeigt, das Rollenspiel mit dem Framework weiter auszubauen. Sollte dies der Fall sein, müsste das generelle Konzept, unter Berücksichtigung der gewonnenen Erfahrungen, noch einmal grundsätzlich überarbeitet werden. Darüber hinaus müssen weitere Aspekte, wie zum Beispiel Sicherheit und Benutzerverwaltung, die bislang noch nicht berücksichtigt wurden, in die Architektur integriert werden. Mit einem Ausbau des Frameworks ist auch eine Erfolgreiche Teilnahme an der „Android Developer Challenge“ denkbar.

Literaturverzeichnis

[Dedaj u. Preisler 2009] DEDAJ, Dennis ; PREISLER, Thomas: *Pervasive Gaming Framework - The World Within*. Projektbericht, HAW Hamburg, 2009

[El-Ayadi 2009] EL-AYADI, Amine: *Pervasive Gaming Framework*. Projektbericht, HAW Hamburg, 2009

[Google] GOOGLE: *Android - An Open Handset Alliance Project*. `http://code.google.com/android/`, Abruf: 19. Juli 2008. – Einstiegsseite der Google Android Homepage

[Hosieny 2009] HOSIENY, Arazm: *Pervasive Gaming Framework*. Projektbericht, HAW Hamburg, 2009

[Hutzler 2009] HUTZLER, Tobias: *Pervasive Gaming Framework*. Projektbericht, HAW Hamburg, 2009

[Kluth 2009] KLUTH, Sascha: *Pervasive Gaming Framework*. Projektbericht, HAW Hamburg, 2009

[Plischka 2009] PLISCHKA, Julia: *Pervasive Gaming Framework*. Projektbericht, HAW Hamburg, 2009