Bachelorarbeit

Stefan Lindenberg

Konzeption und Realisierung eines Frameworks
für mobile Rollenspiele mit Android

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Stefan Lindenberg

Konzeption und Realisierung eines Frameworks für mobile Rollenspiele mit Android

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. rer. nat. Bettina Buth

Abgegeben am 25.12.2009

Stefan Lindenberg

Thema der Bachelorarbeit

Konzeption und Realisierung eines Frameworks für mobile Rollenspiele mit Android

Stichworte

Android, Framework, Mobile Games, Rollenspiel, MMORPG, Pervasive Gaming

Kurzzusammenfassung

Seit drei Jahrzehnten sind Rollenspiele als Gesellschaftsspiel, später dann als Computerspiel für einzelne Spieler und nun in MMORPGs für tausende sich zeitgleich in einer Welt bewegender Spieler, erfolgreich.

In dieser Arbeit werden die Eigenschaften mobiler Rollenspiele durch Verknüpfung der Spielkonzepte der Rollenspiele und der Elemente der Pervasive Games erarbeitet. Es wird geprüft, ob alle Spielelemente bezüglich ihrer Spielbarkeit und der technischen Realisierbarkeit umgesetzt werden können. Basierend auf dem entwickelten Spielkonzept wird ein Framework für die mobile Plattform Android konzipiert und realisiert. Anschließend wird basierend auf diesem Framework der Prototyp eines mobilen Rollenspiels entwickelt.

Stefan Lindenberg

Title of the paper

Design and implementation of a framework for mobile role-playing games with Android

Keywords

Android, Framework, Mobile Games, role-playing games, MMORPG, Pervasive Gaming

Abstract

For the last three decades role-playing games have been successful, at first merely as a kind of board game, then as single-player computer games and finally as MMORPGs for thousands of players acting simultaneously in one single world.

In this paper, the characteristics of mobile role-playing games are developed by linking the concepts of role-playing games with the elements of pervasive games. The practicability of the game elements, considering their playability and technical feasibility, will be verified. Based on this concept a framework for the mobile platform Android will be designed and implemented. Finally a prototype of a mobile role-playing game, based on this framework, will be put into practice.

Inhaltsverzeichnis

1. Einleitung

Das erste Kapitel 1.1 legt die Motivation dar, die hinter dieser Arbeit steht. Im anschließenden Abschnitt 1.2 werden das Ziel dieser Arbeit und die einzubeziehenden Aspekte definiert. Die Gliederung der Arbeit wird im Unterkapitel 1.3 vorgestellt.

1.1 Motivation

Seit Mitte der 1970er Jahre [Nagel 2007] gibt es Rollenspiele (siehe 2.1) und damit die Möglichkeit, sich nur mit Hilfe von Papier, Würfeln, Stiften und der eigenen Fantasie in fremden Welten zu bewegen. Diese Art der Spiele bezeichnet man im Allgemeinen als Pen & Paper-Rollenspiele. Diese Spielkultur wurde mit steigender Entwicklung der Rechnerarchitekturen auch zunehmend als Spiele-Software umgesetzt. Als einzelner Spieler[1] kann man dort, wie auch in den Pen & Paper-Rollenspielen, in einer meist mittelalterlichen Fantasywelt einen Charakter[2] spielen. Durch das Sammeln von Ausrüstungsgegenständen sowie das Erlernen neuer Fertigkeiten, z.B. im Kampf, gewinnt dieser Charakter an Stärke, bis er schließlich in der Lage ist, einen finalen Kampf gegen einen Endgegner zu gewinnen. Dabei besteht in diesen, im Regelfall für nur einen Spieler ausgelegten Spielen, nur Kontakt mit denen von Scripten oder künstlichen Intelligenzen gesteuerten Nicht-Spieler-Charakteren (NPC engl. für Non-Player-Character). Mitte der 1990er Jahre [AchPieSim 2008] wurde mit der Entwicklung grafischer Online-Rollenspiele und der damit gegebenen Möglichkeit, mit tausenden Gleichgesinnten simultan in ein und derselben Welt zu spielen, eine weitere Stufe in der Entwicklung digitaler Rollenspiele erreicht.

Mit der steigenden Bandbreite und den sinkenden Kosten in mobilen Netzen, aber auch der zunehmenden Leistungsfähigkeit der mobilen Endgeräte sowie dem Erscheinen der Plattform

[1] In dieser Arbeit wird der Einfachheit halber bei Personenbezeichnungen ausschließlich die männliche Bezeichnung benutzt. Diese gelten selbstverständlich ebenso für das weibliche Geschlecht.

[2] In dieser Arbeit wird der in Rollenspielen übliche Begriff Charakter für die Spielfigur verwendet.

Android[3] und der darin integrierten direkten Zugriffsmöglichkeit auf *GPS*-Daten zur Lokalisierung, stellt sich nun die Frage, wie eine Umsetzung von Rollenspielen für den mobilen Markt unter Betrachtung der Gesichtspunkte von Pervasive Games möglich ist.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist die Konzeption und Realisierung eines Frameworks[4] zur Entwicklung mobiler Online-Rollenspiele mit *Android*. Dabei soll das Framework sowohl die Schnittmenge der auf mobile Rollenspiele übertragbaren Rollenspiel-Elemente abdecken als auch spezielle Eigenschaften aus dem Bereich Pervasive Gaming, besonders die der Ortsabhängigkeit, berücksichtigen. Um die Spielfunktionen der klassischen Rollenspiele auswählen und die speziellen Funktionen für mobile Rollenspiele ermitteln zu können, wird eine Analyse bestehender Konzepte aus Pen & Paper-Rollenspielen, Computer-Rollenspielen und Online-Rollenspielen durchgeführt und auf deren Umsetzbarkeit für mobile Systeme geprüft. Diese Prüfung umfasst die Fragestellung nach Sinn und Zweck der Umsetzung im Sinne des Spielspaßes und der Spielbarkeit, aber auch unter Betrachtung der technischen Realisierbarkeit mit *Android*. Im Anschluss an die Konzeption und Realisierung des Frameworks sollen dessen Funktion sowie die erarbeiteten speziellen Eigenschaften von mobilen Rollenspielen anhand eines Prototyps gezeigt werden.

1.3 Gliederung

Das Kapitel 2 (Grundlagen) enthält konzeptionelle und begriffliche Erklärungen sowie einen historischen Abriss zu Rollenspielen. Außerdem werden die Themen Pervasive Gaming, Frameworks, *GPS*, die Entwicklungsplattform *Android* und die technische Umsetzung von Online-Rollenspielen vorgestellt.
Das Kapitel 3 (Analyse) vereinigt zuerst die Eigenschaften von Rollenspielen und Pervasive Games zu dem neuen Spielkonzept mobile Rollenspiele. Anschließend werden die aus den

[3] Markennamen werden im Fließtest kursiv markiert und im Verzeichnis für eingetragene Warenzeichen zusammengefasst.
[4] Da die Sprache der Informatik englisch ist, wird in dieser Arbeit durchgehend der übliche englische Begriff Framework und nicht die deutsche Übersetzung Rahmenwerk verwendet.

Eigenschaften folgenden Konsequenzen für das Framework-Design herausgearbeitet und deren Portierbarkeit auf mobile Endgeräte sowohl aus technischer als auch aus Sicht der Spielbarkeit untersucht. Hinzugezogen werden bereits abgeschlossene mobile Rollenspiel-Projekte mit Elementen aus dem Bereich Pervasive Gaming.

In Kapitel 4 (Konzeption) wird anhand der in der Analyse erarbeiteten Ergebnisse ein Konzept für die Realisierung des Frameworks mit *Android* entwickelt.

Kapitel 5 (Realisierung des Frameworks) stellt die für die Realisierung des Frameworks verwendete Hard- und Software vor.

In Kapitel 6 (Realisierung eines Spiels mit dem Framework) wird der mit dem zuvor entwickelten Framework realisierte Prototyp eines Spiels und dessen Funktionen vorgestellt.

Kapitel 7 (Zusammenfassung und Fazit) enthält eine Zusammenfassung der erarbeiteten Ergebnisse und zeigt erreichte Ziele und festgestellte Grenzen auf.

2. Grundlagen

In diesem Kapitel werden die begrifflichen und technischen Grundlagen vorgestellt, die im Verlauf dieser Arbeit zur Anwendung kommen.
Im Abschnitt 2.1 werden Rollenspiele vorgestellt. Nach einer allgemeinen Begriffsdefinition, folgt zuerst die ursprüngliche Variante des Spiels ohne Computer, gefolgt vom Live-Rollenspiel. Zum Abschluss dann die Computerspielvarianten als Einzelspielerspiel ohne Internetnutzung und die ausschließlich für den Mehrspielerbetrieb ausgelegten Online-Rollenspiele. Der für diese Arbeit zentrale Begriff Pervasive Games wird im Abschnitt 2.2 erläutert. Die Abschnitte 2.3, 2.4 und 2.5 widmen sich den technologischen Grundlagen von Frameworks, der *GPS*-Technologie und der Entwicklungsumgebung *Android*. Im abschließenden Kapitel 2.6 wird die technische Realisierung von Online-Rollenspielen vorgestellt. Der Fokus liegt hierbei auf der verwendeten Netzwerkarchitektur.

2.1 Rollenspiele

Dieses Unterkapitel erklärt den Begriff Rollenspiel, soweit er für diese Arbeit relevant ist. Zuerst allgemein, anschließend dann in den Varianten als Spiel ohne Computer (Pen & Paper), als Live-Rollenspiel (LARP), als Einzelspieler-Computer-Rollenspiel und als Online-Rollenspiel (MMORPG für *engl*. Massively Multiplayer Online Role Playing Game).

2.1.1 Begriff des Rollenspiels

Das Fachlexikon der sozialen Arbeit [FdsA 2002] trennt den Begriff Rollenspiel in drei Gruppen.

- Die erste Gruppe erfasst das Rollenspiel als pädagogische Methode zur Konfliktlösung sowohl in der Schule, der Erwachsenenbildung und der Sozialarbeit. Realitätsnahe Situationen werden (oft in Fremdrollen) nachgespielt und anschließend in Gruppen besprochen, um Lösungsmöglichkeiten zur Konfliktbewältigung zu finden.

- Die zweite Gruppe beschreibt die meist von kleinen Kindern gespielten spontanen Rollenspiele (Imitationsspiele), in denen sie die ihnen bekannte Realität meist stark typisiert reproduzieren und sich so an sie anpassen. Beispiele hierfür sind „Vater-Mutter-Kind“ oder „Räuber und Gendarm“.
  Diese Gruppe ist nach [FdsA 2002] im Gegensatz zur ersten tatsächlich ein Spiel.
- Die dritte Gruppe ist die des Rollenspiels als Gesellschaftsspiel. Im Englischen als „Role playing game“ (RPG) bezeichnet, wird dort der Unterschied zu den pädagogischen Rollenspielen bereits in der Wortbedeutung deutlich, die dort nur „Role playing“ heißen.

Für diese Arbeit ist nur die dritte Gruppe relevant. Beschreibungen zu speziellen Ausprägungen von Rollenspielen als Gesellschafts- und Computerspiel finden sich in den folgenden Abschnitten.

2.1.2 Pen & Paper-Rollenspiele

In diesem Abschnitt werden zuerst die Geschichte und dann die Eigenschaften der als „Pen & Paper-Rollenspiele“ bezeichneten Gesellschaftsspiele beschrieben.

2.1.2.1 Geschichte der Pen & Paper-Rollenspiele

Dieses Kapitel beruht solange nicht anders angegeben im Wesentlichen auf [Nagel 2007] und [Wachholz 2005].

Die Ursprünge des allgemeinen Rollenspieles können laut Wachholz bereits im Schachspiel gefunden werden [Wachholz 2005]. Dieses wurde ca. 500 – 100 Jahre v. Chr. in Indien erfunden [Brockhaus 1992]. Die beiden gegnerischen Spieler übernehmen dabei die Rolle eines Feldherrn, dessen reale Soldaten und Untertanen in Spielfiguren mit festgelegten Eigenschaften verwandelt werden.
Mit dem Erscheinen der amerikanischen Taschenbuchausgabe von J.R.R. Tolkiens „Der Herr der Ringe“ 1965, einem Fantasyroman, der insbesondere in amerikanischen College-Kreisen schnell große Popularität erlangte, wurde eine Renaissance der Fantasyliteratur in den USA eingeleitet.
Da sich War-Games oder auch KoSims genannte Konflikt-Simulations-Spiele, die im Prinzip nichts anderes als eine Weiterentwicklung des Schachspiels sind, in den USA bereits großer

Beliebtheit erfreuten, wurde auch diese Szene durch das Erscheinen des „Herrn der Ringe“ verändert. Plötzlich traten in den großen ursprünglich meist historischen Schlachten auch Fantasyfiguren wie Elfen, Orks und Hobbits gegeneinander an. Daraus entwickelten sich Anfang und Mitte der 1970er Jahre erweiterte Spielformen, in denen die Spielfiguren zunehmend individuellere Fertigkeiten bekamen und einige Spiele, die durch einen Spielleiter geleitet wurden.

Im Januar 1974 erschien dann im Verlag TSR in den USA mit *Dungeons & Dragons® (D&D)* das erste Fantasyrollenspiel. Noch heute zählt der Ende der 1970 Jahre in seiner ersten Auflage erschienene Nachfolger *Advanced Dungeons & Dragons®* (AD&D) zu den weltweit meistverkauften Rollenspielen.

In Deutschland begann sich besonders nach der Veröffentlichung der ersten Ausgabe von *Das Schwarze Auge (DSA)* (*Schmidt-Spiele* und *Droemer-Knaur*, 1984), eine eigene Rollenspielszene zu etablieren. Das mittlerweile in der 4. Edition vorliegende Spiel ist das meistverkaufte Rollenspiel in Deutschland.

2.1.2.2 Eigenschaften der Pen & Paper-Rollenspiele

Die in diesem Kapitel dargelegten Aussagen beruhen, solange nicht anders angegeben, im Wesentlichen auf [Janus 2007a] und [Janus 2007b].

Die ursprüngliche Art der Pen & Paper-Rollenspiele wird im Regelfall von drei bis sechs Spielern und einem Spielleiter gespielt. Die Spieler verkörpern je nach gespieltem Spielsystem einen Charakter, der durch Eigenschaften, die größtenteils durch Zahlenwerte ausgedrückt werden, definiert ist. So weiß jeder Spieler, wie die von ihm gespielte Figur aussieht (Größe, Haarfarbe, Augenfarbe etc.), welche Kleidung sie trägt und welche Stärken und Schwächen sie hat. In Abhängigkeit von dem individuellen Regelwerk des Systems, können so nahezu unendlich viele Kombinationen an Charakteren entstehen. Denkbar wäre z.B. ein großer, blonder Ritter, der auf der einen Seite hervorragend mit dem Schwert umgehen kann und so schon einige Drachen erschlagen hat, auf der anderen Seite aber panische Angst vor Spinnen hat.

Neben diesen Spielern mit ihren auf Papier festgehaltenen Charakteren gibt es einen Spielleiter, der den Spielern alle Sinneswahrnehmungen ihrer verkörperten Charaktere und die Aktionen aller anderen Menschen und Tiere beschreibt, welche den Charakteren der Spieler begegnen. Die Spieler wiederum reagieren auf die nur durch die Beschreibungen des

Spielleiters in ihrer Fantasie entstehende Umwelt, indem sie ebenfalls die Handlungen ihrer Charaktere beschreiben oder in geringem Maße (z.B. durch Erheben der Stimme oder Einsetzen von Gesten) schauspielern. Alle Aktionen der Spieler, die keine vorbestimmte Konsequenz haben, z.B. die Frage ob ein Charakter es rechtzeitig schafft, einem fliegenden Messer auszuweichen, werden in den meisten Systemen durch Würfel bestimmt. Die Chance diesem Messer auszuweichen ist dabei höher, je besser die Fertigkeit des Charakters, in diesem Fall im Ausweichen, ist.
Für alle Rollenspiele gibt es Regelbücher, die den Umgang mit solchen Würfelprüfungen, aber auch die Möglichkeiten der Charakterwahl, definieren. Einige Systeme bieten darüber hinaus noch unterschiedlich große Sammlungen an Hintergrundinformationen zu den Kulturen, Gesellschaftsformen, geographischen Gegebenheiten, Pflanzen, Tieren etc. der in dem Spielsystem vorherrschenden Welt an. Damit hat der Spielleiter die Möglichkeit, eine wieder erkennbare Welt zu schaffen.
Rollenspiele können je nach System in jedem denkbaren Szenario spielen. Am häufigsten verwendet werden hier mittelalterliche Fantasy-, Science-Fiction- und Horrorwelten [Wichter 2007].
Im Gegensatz zu herkömmlichen Gesellschaftsspielen haben Rollenspiele weder ein von vornherein festgelegtes Spielende noch ein Spielziel. Im Regelfall reihen sich mehr oder weniger zusammenhängende Abenteuerepisoden aneinander, in denen die Charaktere verschiedene Aufgaben erfüllen und sich so immer neue Fertigkeiten aneignen. Selbst der Tod des Charakters wird im Regelfall durch Erschaffung eines neuen kompensiert. Das Spielziel und damit die Motivation definiert jeder Spieler abhängig von der Wahl seines Charakters selbst: Dem Einen geht es vielleicht um Ruhm und Ehre, dem Anderen um möglichst viel Gold.

2.1.3 Live-Rollenspiel (LARP)

Dieses Kapitel beschreibt die Geschichte und die Eigenschaften eines Live-Rollenspiels.

Live-Rollenspiele, gewöhnlich als LARP (*engl*. für Live Action Role Playing) abgekürzt, sind eine Mischung aus Pen & Paper-Rollenspielen und dem Improvisationstheater, allerdings ohne Zuschauer.

2.1.3.1 Geschichte des Live-Rollenspiels

Jahnke sieht den Anfang des Live-Rollenspiels im Jahr 1991 mit dem Draccon I, der ersten deutschen LARP-Con[5] [Jahnke 2009]. Das Live-Rollenspiel soll neben den offensichtlichen Einflüssen der Pen & Paper-Rollenspiele, auch auf die Reenactments (Wiederaufführung historischer Ereignisse) zurückzuführen sein. Diese sind bis in das Römische Reich zurückverfolgbar. Noch heute gilt die *Landshuter Hochzeit*[6] (seit 1903) neben den historischen Darstellungen des amerikanischen Bürgerkrieges als eines der größten Reenactments der Welt.

2.1.3.2 Eigenschaften des Live-Rollenspiels

Im Unterschied zu Pen & Paper-Rollenspielen werden Live-Rollenspiele nicht am Spieltisch, sondern in realen der Spielhandlung entsprechenden Umgebungen wie Burgen oder Wäldern gespielt. An solchen Cons nehmen je nach Größe der Veranstaltung zeitgleich einige Dutzend bis einige hundert Personen teil. Auf den meist über mehrere Tage andauernden Veranstaltungen schlüpft jeder Spieler vollständig in die Rolle eines Charakters seiner Wahl. Alle Spieler tragen aufwendige Trachten oder Rüstungen und ihrer Rolle entsprechende spezielle Waffen (nicht scharf, oft aus Schaumstoff o. Ä.). Wie in Pen & Paper-Rollenspielen wird von einer Gruppe von Spielleitern eine Geschichte präsentiert, deren Handlung von eingewiesenen Spielern dargestellt wird.

Die Eigenschaften des Live-Rollenspiels sind denen des Pen & Paper-Rollenspiels sehr ähnlich. Die Unterschiede sind hauptsächlich: weniger komplexe Regeln, eine viel größere Zahl zeitgleich teilnehmender Spieler und die Verschmelzung des Spielers mit der eigenen Spielfigur. Der Spieler ist sein Charakter und spielt spontan wie im Improvisationstheater seine Rolle.

2.1.4 Computer-Rollenspiele

Dieses Kapitel beschäftigt sich mit den Einzelspieler-Computer-Umsetzungen der Rollenspiele. Diese werden in dieser Arbeit als Computer-Rollenspiele bezeichnet.
Die als MMORPG bekannte Mehrspieler-Online-Variante wird in Kapitel 2.1.5 behandelt.

[5] „Con“ (engl. für Convention) ist die gebräuchliche Bezeichnung für LARP-Veranstaltungen.
[6] Die Landshuter Hochzeit ist ein Fest, das alle vier Jahre im Sommer in Landshut zur Erinnerung an die im Jahre 1475 in Landshut stattgefundene Heirat des bayerischen Herzogs Georgs des Reichen mit Hedwig Jagiellonica aufgeführt wird [vgl. Wikipedia 2009b].

2.1.4.1 Geschichte der Computer-Rollenspiele

Dieses Kapitel beruht soweit nicht anders angegeben auf [Barton 2007a], [Barton 2007b] und [Barton 2007c].

Die Geschichte der Computer Rollenspiele lässt sich bis in das Jahr 1974, in dem auch das erste Pen & Paper-Rollenspiel *Dungeons & Dragons®* (D&D) erschien (siehe 2.1.2.1), zurückverfolgen. Zwei Programmierer aus Illinois entwickelten dnd (auch The Game of Dungeons) (siehe Abbildung 2-1) für das Großrechnersystem PLATO. Das Spiel dnd (Gary Whisenhunt, Ray Wood, 1974) gilt als das älteste noch erhaltene Computer-Rollenspiel, wohl aber nicht als das älteste jemals entwickelte. Noch älter ist pedit5, welches jedoch damals konsequent von den Administratoren der Großrechner gelöscht wurde und somit heute nicht mehr erhalten ist. Es wird als wahrscheinlich angesehen, dass es zuvor noch rein textbasierte Rollenspiele gab, konkrete Informationen darüber sind jedoch ebenfalls verloren gegangen.

In den 35 Jahren, die seit der Entwicklung von dnd vergangen sind, haben die Computer-Rollenspiele im Bereich Grafik, Benutzerführung und Steuerung eine bemerkenswerte Entwicklung durchlaufen. Heutige Bestseller wie *Fallout® 3* und *Das schwarze Auge: Drakensang* bieten hochauflösende 3D-Grafik und ausgefeilte Steuerungen mit Maus und Tastatur.
Die Kernelemente des Genres, die ein Rollenspiel ausmachen, sind aber noch die gleichen wie in den Anfängen. Details dazu im folgenden Kapitel (2.1.4.2).

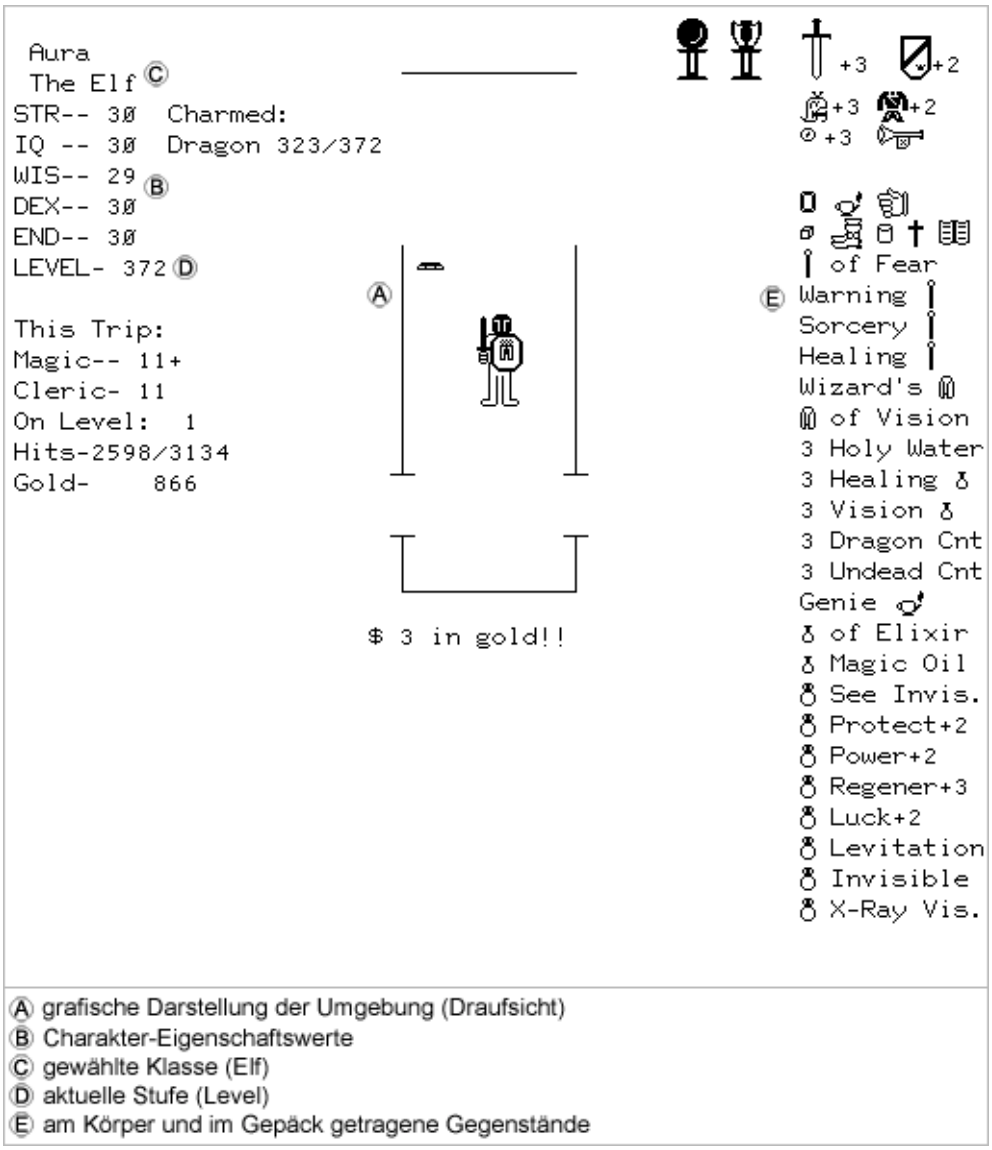

Abbildung 2-1: dnd, ältestes noch erhaltenes Computer-Rollenspiel, 1974 [Wikipedia® 2009a]

2.1.4.2 Eigenschaften der Computer-Rollenspiele

Dieses Kapitel beruht soweit nicht anders angegeben auf [Barton 2007a], [Barton 2007b], [Barton 2007c] und [AchPieSim 2008].

Im Folgenden werden die für ein Computer-Rollenspiel typischen Eigenschaften, die sich im Laufe der Jahre durchgesetzt haben, zusammengefasst und ihr Vorkommen mit dem in Pen & Paper-Rollenspielen verglichen. Nicht alle diese Funktionen müssen zwangsweise in einem Computerspiel integriert sein, damit es sich um ein Rollenspiel handelt. Einige der Funktionen schließen sich sogar gegenseitig aus. Zusätzlich wird bei einzelnen Eigenschaften

sowohl deren erste als auch eine beispielhafte heutige Verwendung in einem Computer-Rollenspiel aufgeführt.

Spielfigur

Solo-Charakter

Die aus Pen & Paper-Rollenspielen bekannte Darstellung eines Charakters pro Person ist auch in den Computer-Rollenspielen weit verbreitet. Der Spieler hat eine Spielfigur, die ihn durch das gesamte Spiel begleitet.

Erstmals: dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *Gothic™ 3*, *Piranha Bytes*, 2006

Gruppe von Charakteren

Eine in Pen & Paper-Rollenspielen kaum anzutreffende Variante ist eine Gruppe von mehreren Charakteren pro Spieler. Damit kann der Spieler aufgrund der breiteren Auswahl an Charakteren in seiner Gruppe (z.B. ein Kämpfer, ein Zauberer und ein Heiler) und aufgrund der breiteren Menge an Fertigkeiten besser auf die zu lösenden Aufgaben in der Spielwelt reagieren.

In einer Variante dieses Systems kann ein Hauptcharakter Begleiter anwerben oder anheuern, welche diesen für eine gewisse Zeit durch das Spiel begleiten und dabei in den meisten Spielen vom Spieler gesteuert werden können. Diese Variante hat sich zunehmend durchgesetzt und findet heutzutage meistens dann Anwendung, wenn nicht nur ein Charakter gespielt wird.

Erstmals: Dungeon, Don Daglow, 1975, für PDP-10

Heute (Variante, s. o.): *Das schwarze Auge: Drakensang*, Radon Labs, 2008, (siehe Abbildung 2-2)

Charakterentwicklung

Bestandteile

Klassen und/oder Rassenwahl

Wie auch in den Pen & Paper-Rollenspielen kann die Charakterentwicklung in einigen Computer Rollenspielen durch die Wahl der Klasse (Professionen wie Krieger, Magier, Dieb o. Ä.) und der Rasse (Mensch, Zwerg, Elf o. Ä.) zu Spielbeginn vorbestimmt werden. Diese Auswahl bestimmt Startwerte oder Grenzen der Charakter-Eigenschaftswerte und Fertigkeiten (s. u.).

Erstmals: *Akalabeth*™: World of Doom, Richard Garriott, 1979, Apple II

Heute: *The Elder Scrolls® IV*: *Oblivion*™, Bethesda Softworks, 2006

Charakter-Eigenschaftswerte

Analog zu Pen & Paper-Rollenspielen, werden in vielen Computer-Rollenspielen die Eigenschaften Mut, Intelligenz, Kraft etc. durch Zahlenwerte ausgedrückt.

Erstmals: dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *Das schwarze Auge: Drakensang*, Radon Labs, 2008

Fertigkeiten

Zusätzlich können Charaktere besondere Fertigkeiten wie Schleichen, Schlösserknacken, Sprechen spezieller Zauber o. Ä. haben. Auch dieses Spielelement findet sich in Pen & Paper-Rollenspielen.

Erstmals: nicht eindeutig ermittelbar

Heute: *Das schwarze Auge: Drakensang*, Radon Labs, 2008

Verbesserung des Charakters

Um die Fertigkeiten und Eigenschaften eines Charakters zu verbessern, finden sich verschiedene Systeme. Diese treten auch häufig gemeinsam auf.

Lernen nach Stufenaufstieg

Das bekannteste System ist das des Stufenaufstieges. Charaktere erhalten für in der Spielwelt vollbrachte Taten (Lösen von Aufträgen, Besiegen von Gegnern) Erfahrungspunkte. Nach Erreichen einer bestimmten Menge an gewonnener Erfahrung steigt der Charakter einen Level auf. Somit kann der Spieler Eigenschaften oder Fertigkeiten seines Charakters anheben. Der Charakter-Level ist damit ein Gradmesser für die Stärke des Charakters in unterschiedlichen Spielsituationen. Dieses System findet auch sehr häufig Verwendung in Pen & Paper-Rollenspielen.

Erstmals: dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *Das schwarze Auge: Drakensang*, Radon Labs, 2008

Lernen durch Anwendung

Einen anderen Ansatz bietet das Lernen durch Anwendung. Hierbei steigen die Fertigkeiten eines Charakters, je häufiger er diese anwendet. Die Fertigkeit, ein Schloss zu knacken, steigt also nur dann, wenn es auch immer wieder probiert wird. Dieses System findet im Pen & Paper-Rollenspielbereich kaum Verwendung.

Erstmals: nicht eindeutig ermittelbar

Heute: *The Elder Scrolls® IV*: *Oblivion™*, Bethesda Softworks, 2006

Lernen durch Trainer

Eine weitere Möglichkeit, die häufig als Ergänzung eingesetzt wird, ist das Lernen durch einen Trainer. Der Spieler kann durch Abgabe von Gold o. Ä. bei Trainern bzw. Lehrern, die er in der Spielwelt finden kann, seine Eigenschaften und Fertigkeiten verbessern. Zum Teil ist dieser Vorgang noch an eine vorherige Erfüllung von Aufträgen gebunden, die der Trainer dem Charakter stellt.

Erstmals: nicht eindeutig ermittelbar

Heute: *Gothic™ 3, Piranha Bytes*, 2006

Verbessern durch das Finden von Gegenständen

Eine Funktion, die von Beginn an jedes Rollenspiel beinhaltete, ist das Verbessern der Charakter-Fertigkeiten durch das Finden von Gegenständen. Das können Ringe sein (z.B. Kraftringe, Unsichtbarkeitsring), jede Art von Rüstungsteilen und natürlich Waffen (Beispiel für einen Gegenstand siehe Abbildung 2-3).

Erstmals: dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *The Elder Scrolls® IV: Oblivion*™, Bethesda Softworks, 2006

Spielwelt

Szenario

Nicht nur im Pen & Paper-Bereich, sondern auch im Bereich der Computer-Rollenspiele, ist das Fantasyszenario am weitesten verbreitet. Fast alle diese Welten verwenden mehr oder weniger viele Elemente aus dem Roman „Der Herr der Ringe“ von J.R.R. Tolkien. Typisch dafür sind Zwerge und Elfen und als Feinde mächtige Zauberer. Andere Szenarien sind z.B. Endzeit und Science-Fiction.

Örtlichkeit

Die ersten Rollenspiele, sowohl am PC als auch als Pen & Paper-Spiel, spielten weitestgehend in so genannten Dungeons (aus dem Englischen: Kerker). Der Charakter sah während des Spiels das Tageslicht höchstens, um gefundene Gegenstände zu verkaufen und neue Ausrüstung einzukaufen.

Die oft endlos scheinenden Gewölbe wurden über viele Ebenen immer tiefer, von Ebene zu Ebene wurden die Gegner (auch: Monster) immer grausamer und vor allem immer stärker und irgendwo ganz unten im Verließ wartete der Endgegner nur darauf, dass ein unerschrockener Held sich mit ihm messen wollte.

Heutzutage spielen viele Computer-Rollenspiele auch oberirdisch in Dörfern, Städten, Wäldern etc., dort stehen oft komplexere in sich logische Geschichten weiter im Vordergrund, ohne aber dass das Töten feindseliger Wesen unwichtiger geworden wäre. Es handelt sich hierbei jedoch nicht mehr um das zentrale Spielelement.

Sowohl damals wie auch heute arbeiten einige Spiele, deren Hauptaugenmerk weniger auf der Geschichte, sondern mehr auf dem Bekämpfen von Monstern liegt, mit zufällig generierten Dungeons.

Sichtbereich

Schon früh wurde es in Computer-Rollenspielen üblich, dass die Kartenabbildungen der Gebiete auf denen man sich gerade bewegt, nicht vollständig sichtbar sind. Sichtbar ist immer nur der Bereich, in dem sich die Spielfigur bereits befand oder sich im Augenblick befindet.

Erstmals: Dungeon, Don Daglow, 1975, für PDP-10

Heute: *The Elder Scrolls® IV: Oblivion*™, Bethesda Softworks, 2006

Monster

In vielen Computer-Rollenspielen gibt es unzählige Gegner, die keine andere Aufgabe haben, als für den Spieler zeitgleich sowohl ein Hindernis als auch Trainingspartner zu sein. Diese werden üblicherweise als Monster bezeichnet. Durch das Besiegen dieser Monster gewinnen die Spieler Erfahrungspunkte oder finden hilfreiche Gegenstände, um so ihren Charakter zu verbessern. Je nach Spiel können dies Tiere, Fabelwesen aber auch Menschen sein.

Nicht-Spieler-Charaktere

Alle im Spiel anzutreffenden NPCs, ob freundlich oder feindlich gesinnt, werden von der Software gesteuert. Dies führt im Vergleich zu Pen & Paper-Rollenspielen dazu, dass die Spielwelt und ihre Charaktere nur so flexibel reagieren können, wie die Software es zulässt und damit z.B. Dialoge mit NPCs nur über eine Auswahl an Antwortmöglichkeiten erfolgen.

Spielziel

Endgegner

Im Gegensatz zu Pen & Paper-Rollenspielen ist das Spielziel bei Computer-Rollenspielen klar definiert. Es gibt in den meisten Spielen einen Bösewicht, der vom Spieler unschädlich gemacht werden muss. Je nach epischer

Breite der Geschichte bedroht dieser Bösewicht ein Dorf, eine ganze Region oder sogar die ganze (Spiel-) Welt. Ist diese Aufgabe gelöst, ist das Spiel auch zu Ende.

Quests

Der Begriff Quest (aus dem Englischen für *Suche*) ist im Bereich der Computer-Rollenspiele eine übliche Bezeichnung für zu lösende Aufgaben, die vom System gestellt werden. Diese sind in den meisten Spielen nicht zufällig generiert, sondern statisch in die Geschichte eingebaut. Dabei kann es sich um so genannte Nebenquests handeln. Diese haben mit dem primären Spielziel nichts zu tun, liefern aber bei Erfüllung ansprechende Belohnungen und machen die erzählte Geschichte lebendiger. Es können aber auch Quests sein, die vom Spieler gelöst werden müssen, um später das Spielziel zu erreichen, d.h. den Hauptquest zu lösen.

Erstmals (nur ein Hauptquest): dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *Das schwarze Auge: Drakensang*, Radon Labs, 2008

Kampfsystem

Rundenbasiert

Der Kampf gegen Gegner aller Art ist in vielen Rollenspielen ein zentrales Element. Die rundenbasierte Variante, diese Kämpfe mit all den Sonderfunktionen – wie z.B. Zaubersprüche – im Computerspiel abzubilden, ist dem Schach ähnlich. Die Aktionen der am Kampf beteiligten Figuren wie angreifen, zaubern etc. werden meist rundenweise nacheinander durchgeführt und aus der Vogelperspektive gezeigt.

Diese Variante kommt in der beschriebenen ursprünglichen Version heute kaum noch zum Einsatz. Stattdessen läuft der Kampf fließend weiter, das heißt es passiert auch dann etwas, wenn man als Spieler nur zuschaut. Das Spiel kann aber jederzeit pausiert werden, so dass den Charakteren in Ruhe neue Befehle gegeben werden können.

Erstmals: dnd, Gary Whisenhunt, Ray Wood, 1974, für PLATO

Heute: *Das schwarze Auge: Drakensang*, Radon Labs, 2008

Echtzeit

Eine erst mit steigender Rechnerleistung aufgekommene Variante ist die des Kampfes in Echtzeit. Das Spiel wird also nicht pausiert und der Spieler entscheidet weniger mit seinen strategischen Fertigkeiten, als mit seiner Geschicklichkeit mit Maus und Tastatur den Kampf.

Erstmals: Telengard, Daniel Lawrence, Avalon Hill, 1982, für Commodore *PET*

Heute: *Gothic™ 3*, *Piranha Bytes*, 2006

Grafik

Allgemein

Der auffälligste Unterschied zwischen Computer-Rollenspielen und Pen & Paper-Rollenspielen ist die Existenz einer grafischen Darstellung. Bei Computer-Rollenspielen sieht der Spieler am Monitor, was seine Charaktere sehen. Die optische Wahrnehmung der Charaktere muss sich nicht mehr in der Fantasie des Spielers abspielen, sondern wird auf dem Bildschirm dargestellt.

Es haben sich annähernd parallel zwei Varianten der Darstellung von Computer-Rollenspielen herausgebildet: die Vogel- und die so genannte Ego-Perspektive.

Vogelperspektive

Die klassische Draufsicht wurde mit steigender grafischer Rechenleistung der Computersysteme durch eine isometrische Vogelperspektive abgelöst. Heute gibt es sogar vollständig dreidimensional drehbare Ansichten.

Ego-Perspektive

Die Alternative dazu ist die Ansicht entweder aus den Augen des Charakters oder kurz hinter ihm. In beiden Fällen sind diese Ansichten heutzutage vollständig dreidimensional. Der Spieler kann also einen Rundumblick einschließlich Blick zum Boden und in den Himmel durchführen.

360° freie 3D-Perspektive

Diese Ansicht ist in gewisser Form eine Mischung der beiden vorherigen und kommt bei Spielen der neusten Generation zum Einsatz. Dem Nutzer sind keinerlei Grenzen mehr gesetzt, er kann die Perspektive in jede Richtung und in jedem Winkel um den Charakter drehen und zoomen (siehe Abbildung 2-2).

Ⓐ grafische Darstellung der Umgebung (360° freie 3D-Perspektive)
Ⓑ Kurzinformationen der eigenen Charaktere (Lebensenergie, Mana, wirkende Effekte)
Ⓒ Übersichtskarte der nahen Umgebung (Minimap)
Ⓓ Charakter-Eigenschaftswerte
Ⓔ gewählte Klasse/Rasse (Söldner/Ambosszwerg)
Ⓕ aktuelle Stufe (Level) und zur Verbesserung des Charakters verfügbare Punkte
Ⓖ am Körper und im Gepäck getragene Gegenstände

Abbildung 2-2: Screenshot: *Das schwarze Auge: Drakensang*, oben: Spielansicht, unten: Charakterübersicht

2.1.5 Online-Rollenspiele

Dieses Kapitel und seine Unterkapitel beruhen soweit nicht anders angegeben auf [AchPieSim 2008], [Poitzmann 2007], [Kent 2003] und [Leyde 2007].

Dieses Kapitel beschäftigt sich mit den Online-Rollenspielen (MMORPG). In Abschnitt 2.1.5.1 wird zunächst die Geschichte der Online-Rollenspiele beschrieben, im folgenden Abschnitt 2.1.5.2 dann deren besondere Eigenschaften speziell im Unterschied zu denen der Computer-Rollenspiele.

2.1.5.1 Geschichte der Online-Rollenspiele

Die grafischen Online-Rollenspiele, in der Form wie sie heute millionenfach gespielt werden (der Marktführer *World of Warcraft*™ zählte im Dezember 2008 weltweit 11,5 Millionen Abonnenten [Blizzard 2008]), finden ihren Ursprung wie auch die Computer-Rollenspiele in den Pen & Paper-Rollenspielen. Sie sind keine Weiterentwicklung der Computer-Rollenspiele, sondern haben sich parallel entwickelt aus den so genannten MUDs (Multi User Dungeons, Multi User Dimensions oder auch Multi User Domains – in dieser Arbeit wird die gebräuchlichste Variante Multi User Dungeon verwendet). Das erste heute noch nachweisbare MUD ist MUD-1 aus dem Jahr 1977. Das Spielprinzip ähnelte dem des Pen & Paper-Rollenspiels *Dungeons & Dragons*® und war vollständig textbasiert. Das Prinzip war das gleiche wie bei Computer-Rollenspielen: man spielte einen Charakter, der Dungeons nach Schätzen durchforschte, so Erfahrungspunkte ansammelte und immer stärker wurde. Der entscheidende Unterschied dazu war, dass man über Netzwerke mit anderen Spielern zusammen spielen konnte, das aber nicht musste. Denn niemand wurde daran gehindert auch Mitspieler anzugreifen. Die Kommunikation fand über simple Chat-Systeme statt und im Gegensatz zu Computer-Rollenspielen gab es kein klares Spielziel (siehe auch 2.1.5.2). Wenn sich Spieler in diesen virtuellen Gemeinschaften auf irgendeine Weise besonders hervorhoben, konnten sie auch vollständig oder nur teilweise die Seite wechseln und helfen, die Welt des MUD auszubauen, in dem weitere Orte (der Begriff Ort kann hier auch für einen Wald, eine Höhle o. Ä. stehen) und deren Besonderheiten hinzugefügt werden. Diese Personen übernehmen damit in gewisser Form die Funktion des Spielleiters in Pen & Paper-Rollenspielen.

Die MUD-Szene hält sich bis heute, es werden sogar noch weitere nicht kommerzielle Projekte entwickelt[7].

Mit den verbesserten grafischen Möglichkeiten und den schnelleren Internetverbindungen folgte dann 1996 das erste grafische Online-Rollenspiel *Meridian59*™[8] (3DO, 1996). Wer sich gegen eine monatliche Gebühr einen Account für das Spiel anlegte, bekam die Möglichkeit mit bis zu 180 Spielern gemeinsam mit- und gegeneinander zu spielen, zu chatten und sich im Duell mit anderen Charakteren zu messen. In einem kleinen Fenster konnte man die Umgebung des Charakters aus der Ego-Perspektive sehen, Monster und Charaktere anderer Spieler angreifen, mit ihnen handeln, ihnen zuwinken und sogar mit ihnen tanzen. Spielfunktionen, die auch heute noch in den meisten Online-Rollenspielen gängig sind. Die Server von Meridian fassten bis zu 250 Spieler - eine Zahl, die auf den ersten Blick groß erscheint, sich aber nur in der Größenordnung bewegt, in der auch MUDs angesiedelt waren[9]. Bereits etwa ein Jahr später mit der Veröffentlichung von *Ultima Online*™ (Origin, 1997) verzehnfachte sich dieser Wert. *Ultima Online*™ ist das erste kommerziell erfolgreiche Online-Rollenspiel, im Gegensatz zu Meridian sah man dort die Umgebung aus einer isometrischen 3D-Perspektive. In den kommenden Jahren bis heute sind weltweit etliche verschiedene Online-Rollenspiele mit zunehmend besserer Grafik, verbesserter Steuerung, verschiedenen Szenarien und Hintergründen erschienen. Der Markt wird deutlich dominiert von *World of Warcraft*™ (*Blizzard Entertainment*®, 2004), jährlich erscheinen jedoch weitere konkurrierende Titel.

2.1.5.2 Besondere Eigenschaften der Online-Rollenspiele

Im Vergleich zu Computer-Rollenspielen liegt der Unterschied der Online-Rollenspiele vor allem in der Tatsache, dass ein Spieler zeitgleich mit oft tausenden anderen Spielern parallel in der gleichen Welt spielt. Diese grundlegende Eigenschaft der Online-Rollenspiele bringt neue Eigenschaften mit sich und modifiziert z.T. bereits bekannte Eigenschaften der Computer-Rollenspiele (vgl. 2.1.4.2).

[7] Unter http://mudder.info finden sich Links zu noch aktiven MUDs.
[8] Ende Dezember 2009 soll Meridian59 nach 13 Jahren endgültig eingestelllt werden [Meridian 2009].
[9] Dies ist der Grund, warum einige Fachleute Ultima Online und nicht Meridian als erstes Online-Rollenspiel sehen, da Meridian im Prinzip nur ein grafisches MUD sei.

Spielfigur

In Online-Rollenspielen tritt der ursprüngliche Gedanke des Spielens einer Rolle wieder stärker in den Vordergrund. Damit wird im Großteil der Spiele pro Spieler nur ein Charakter zeitgleich vertreten. Es ist jedoch nicht unüblich, dass Spieler verschiedene Typen von Charakteren haben und diese abwechselnd spielen.

Charakterentwicklung

Die Prinzipien der Charakterentwicklung in Online-Rollenspielen unterscheiden sich nicht von denen in Computer-Rollenspielen. Auch hier liegt ein besonderes Augenmerk in allen Spielen auf der Verbesserung der Charaktere durch das Finden seltener Gegenstände beim Töten von computergesteuerten Gegnern (Beispiel siehe Abbildung 2-3).

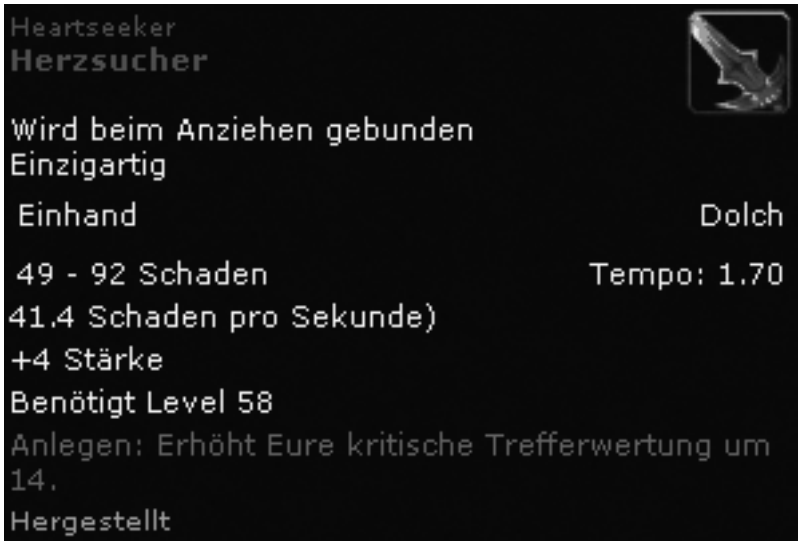

Abbildung 2-3: Screenshot der Spielwerte eines Dolches aus dem Spiel *World of Warcraft*™

Spielwelt

Die Aussagen zur Spielwelt in Computer-Rollenspielen gelten ebenso für Online-Rollenspiele, werden aber durch weitere Punkte ergänzt:

Persistenz

Die Spielwelten von Online-Rollenspielen werden als persistente (anhaltende) Welten bezeichnet, da sie für den Spieler zu jeder Zeit zugängig sind und die Zeit in der Welt auch dann weiterläuft, wenn ein Spieler nicht in das Spiel eingeloggt ist. Es geschehen also Ereignisse in seiner Abwesenheit. Als Konsequenz daraus ergibt sich, dass beim Ausloggen aus einem Online-Rollenspiel der aktuelle Charakter auf dem Server des Spielanbieters gespeichert wird und auch genau diese Version des Charakters zu Beginn der nächsten Spielsitzung wieder geladen wird. Im Gegensatz zu

Computer-Rollenspielen kann man also nicht das Spiel speichern, eine Handlung durchführen, entscheiden dass man die Handlung wiederholen möchte (weil der Versuch z.B. gescheitert ist) und das Spiel neu laden. Jede vom Spieler in der persistenten Welt durchgeführte Handlung hat sofortigen und unwiderruflichen Einfluss auf die Spielwelt.

Nicht-Spieler-Charaktere

Hier gelten die gleichen Aussagen wie für Computer-Rollenspiele.

Allerdings werden die vom Spieler getöteten NPCs in Online-Rollenspielen im Gegensatz zu Computer-Rollenspielen nach einiger Zeit (in den meisten Fällen wenige Minuten) wieder belebt. Das Ziel ist immer, dass jeder Spieler zumindest in Bezug auf die von der Software gestellte Spielwelt, immer das gleiche Spielerlebnis hat. Dies gilt z.B. in *World of Warcraft*™ sowohl für von anderen Spielern getötete NPC-Händler, die auch danach noch zur Verfügung stehen sollen, als auch für Monster, die einfach nur die Aufgabe haben, den Charakteren durch besiegen die Möglichkeit zur Erfahrungsgewinnung zu geben.

Instanzen

Einige Online-Rollenspiele wie auch *World of Warcraft*™ verwenden so genannte Instanzen. Eine Instanz beschreibt eine Kopie eines Dungeons. Auf diese Art und Weise kann eine Gruppe von Spielern (bei *World of Warcraft*™ bis zu 40 Spieler) ohne Kontakt mit anderen Gruppen Quests lösen und versuchen, spezielle Gegner zu besiegen. Diese Funktion soll vor Problemen wie z.B. „Kill Stealing“[10] und „Spawn Camping“[11] schützen [Blizzard 2005].

Handelssystem

Im Gegensatz zu Computer-Rollenspielen gibt es in MMORPGs ein eigenes Wirtschaftssystem. Da der Handel zwischen den von Menschen gesteuerten Charakteren abläuft, basiert dieser größtenteils auf Angebot und Nachfrage.

[10] Da nur die Gruppe von Spielern die Erfahrungspunkte und die von den Gegnern fallen gelassenen Gegenstände bekommt, die den letzten tödlichen Treffer gemacht hat, können so außerhalb von Instanzen Erfahrungspunkte und Gegenstände gestohlen werden.

[11] Wenn viele Spieler zeitgleich den gleichen Quest lösen wollen, der als Auftrag hat, einen bestimmten Gegner zu töten, warten diese darauf, dass dieser nach seinem letzten Tod vom System neu erzeugt wird. Dieses Warten ist vergleichbar mit dem Anstehen an einer Supermarktkasse.

Spielziel

Endgegner und Quests

Online-Rollenspiele haben im Regelfall keinen klar definierten Endgegner eines Handlungsstranges, der von Spielern besiegt werden kann. Die verwendeten Hintergrundgeschichten nehmen zwar epische Ausmaße an, der Spieler bekommt aber nie das Gefühl, das Spiel beendet zu haben. Keine Aufgabe ist so final, dass das Gefühl aufkäme, keinen Grund mehr zu haben, die Spielwelt wieder zu betreten. Das heißt nicht, dass nicht irgendwann der Spieler alle von den Entwicklern in Form von Quests gestellten Aufgaben gelöst hat, jedoch hat die Geschichte noch kein Ende gefunden, der Kampf geht weiter. In *World of Warcraft*™ beispielsweise dreht sich der Großteil der im Spiel erlebten Geschichte um den Kampf zwischen zwei großen Volksgruppen. Unabhängig davon, welche der acht Rassen der Spieler bei der Charaktererschaffung wählt, gehört er immer einer der beiden Gruppen an (Allianz bestehend aus Menschen, Nachtelfen, Gnomen, Zwergen oder Horde bestehend aus Orks, Trollen, Tauren, Untoten). Die gestellten Quests drehen sich oft um die Thematik des Kampfes gegen die feindliche Gruppierung. Es gibt aber keinen Quest, den man lösen kann, um die Gegner vollständig zu besiegen, denn das würde zur Konsequenz haben, dass das Spiel beendet wäre. Würde ein Spieler der Allianz einen möglichen Endquest gegen die Horde lösen und jene damit besiegt haben, wäre die Allianz der Sieger über die Horde. Da es sich um eine gemeinsame Welt mit allen anderen Spielern handelt, müssten konsequenterweise alle Hordenspieler besiegt sein und alle anderen Allianzspieler hätten aus der Spielgeschichte keine bestehende Motivation mehr weiterzuspielen.

Aufgrund dieser Problematik sind Quests in Online-Rollenspielen in den meisten Fällen, wenn sie von einem Spieler gelöst sind, ohne bleibende Auswirkungen auf die Spielwelt und müssen sich von jedem Spieler durchführen lassen. Dies kann zu skurrilen Situationen führen. In *World of Warcraft*™ gibt es z.B. einen Nebenquest, in dem zwei Kinder einen Ring in einen See fallen lassen haben und ihn nun zwischen den angriffslustigen Fischen nicht wieder finden. Nun ist es Aufgabe der Spieler, diesen Ring wieder zu besorgen. Die Möglichkeit, dass nun jeder Spieler diesen Quest durchführen können soll, hat zur Folge, dass diese beiden Kinder nun seit fünf Jahren am Ufer des Sees sitzen und sicherlich bereits einige Millionen Mal den Ring verloren haben.

So wie die Kinder immer wieder den gleichen Ring verlieren, belebt die Software auch alle von den Spielern getöteten Questgegner wieder. Das Töten einer Räuberbande in *World of Warcraft*™ wiederholt sich also auch ständig (s. o. Nicht-Spieler-Charaktere). Solche Probleme tauchen in Computer-Rollenspielen nicht auf.

Kampfsystem

In fast allen Online-Rollenspielen finden die Kämpfe in Echtzeit statt. Somit kann kein Spieler einen Kampf pausieren, denn damit würde die Welt auch für alle anderen Mitspieler stillstehen. Allerdings sind die zentralen Eigenschaften der Kämpfe im Allgemeinen Glück und Taktik. Geschicklichkeit spielt in Kämpfen in Online-Rollenspielen nur eine untergeordnete Rolle.

Kampf gegen andere Spieler

Eine Frage die sich in Computer-Rollenspielen nicht stellt, ist die Frage nach dem Kampf gegen andere Spieler.

In Online-Rollenspielen geht die Bandbreite an angebotenen Spielen über MMORPGs wie *Guild Wars*® (*ArenaNet*, 2005), deren Kernkomponente das so genannte PvP (für *engl*. Player versus Player) ist [GuildWars 2005], über Spiele wie *World of Warcraft*™, auf denen PvP nur auf bestimmten Servern oder in bestimmten Regionen möglich ist, bis hin zu so extremen Beispielen wie *Everquest*® *2* (Sony Online Entertainment, 2004), die erst aufgrund der Kritik der Spieler überhaupt „Spieler gegen Spieler“-Kämpfe mit einer Erweiterung zugelassen haben [Stolzenberg 2005].

Generell ist dies ein sehr kontroverses Thema, da die Frustration oft sehr hoch ist, wenn ein deutlich stärkerer Spieler den eigenen Charakter von hinten niederstreckt.

Grafik

Hier gelten die gleichen Aussagen wie zu den Computer-Rollenspielen.

Kommunikation

In allen Online-Rollenspielen erfolgt eine Kommunikation zwischen Spielern auf einem spielinternen Chatsystem. Dies ist je nach Spiel entweder lokal begrenzt, so dass eine Kommunikation nur mit den sich räumlich in der Nähe befindenden Spielern möglich ist oder von der Position unabhängig, so dass private Chat-Nachrichten an alle Spieler ortsunabhängig versendbar sind. Dies ist die häufiger verwendete Variante.

2.2 Pervasive Gaming

Pervasive Gaming ist eine auf die Softwareunterhaltungsbranche spezialisierte Form des Pervasive Computing (*engl.:* pervasive = durchdringend) bzw. Ubiquitous Computing[12] (*engl.:* ubiquitous = allgegenwärtig) [Napitupulu 2006]. Laut Mattern beschreibt der Begriff Pervasive Computing die „Allgegenwärtigkeit von Informationsverarbeitung und damit einhergehend den jederzeitigen Zugriff auf Daten von beliebiger Stelle“ [Mattern 2001]. Nach Björk u. a. sind Pervasive Games Spiele, die für den Spieler zu jederzeit gegenwärtig und verfügbar sind [BjöHolLjuMan 2002]. Sie sind ortsunabhängig, können also überall und jederzeit gespielt werden, und können ortssensitiv sein, beziehen also die aktuelle Position des Spielenden ins Spielgeschehen ein. Lankoski u. a. beschreiben Pervasive Games als eine neue Form der digitalen Spiele, die virtuelle und physische Realität vereinen [LanHalNumLahMäyErm 2004]. Abbildung 2-4 zeigt die Stellung der Pervasive Games im Vergleich zu anderen Spielarten, bezogen auf deren Verflechtung zwischen virtueller und reeller Welt und der Allgegenwärtigkeit der Computer.

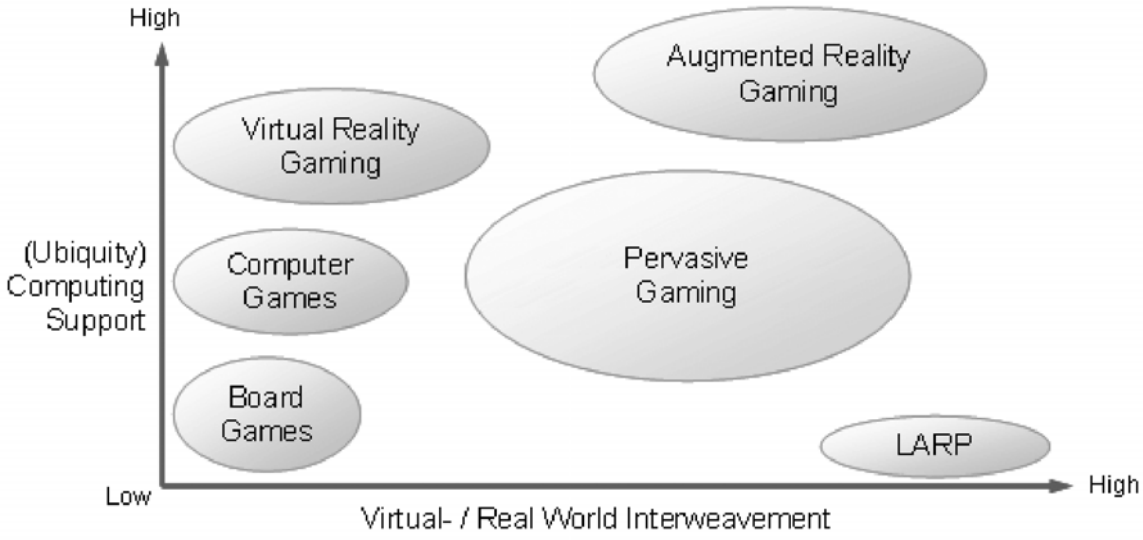

Abbildung 2-4: Pervasive Games und andere Spielarten im Vergleich [OKS 2009]

[12] Der Begriff Ubiquitous Computing bezeichnet laut [Weiser 1991] Computer, die in den Hintergrund treten, nur ein Mittel zum Zweck sind und somit zunehmend aus dem Bewusstsein der Menschen verschwinden.

Praktisch bedeutet dies: die Spieler bewegen sich durch die physikalische Welt, um ihren Charakter in der virtuellen Welt zu bewegen. Das Spiel kann in den verschiedenen Kontexten des Alltagslebens, auch im Privaten zu Hause ausgetragen werden [JegWib 2006].

In einem Vortrag zu Pervasive Games hat Prinz die in Tabelle 2-1 ersichtlichen Unterschiede zu klassischen Computerspielen herausgearbeitet [Prinz 2006].

Klassisches Computerspiel	**Pervasive Game**
Spielort ist der Computer	Keine Beschränkung, Ort/Position des Spielers dient zur Spielsteuerung
Spielwelt existiert virtuell im Computer	Reale Umgebung wird zur Spielwelt
Spieler kontrolliert einen Avatar als Spielhelden	Spieler wird selbst zur Spielfigur
Spieler spielt (meist) alleine	Kooperative Spielformen
Spiel wird für einen bestimmten Zeitraum fokussiert gespielt	Spiel erstreckt sich über einen längeren Zeitraum und ist mit anderen Aktivitäten verwoben
Spieler kontrolliert das Spiel mit speziellen Interaktionsgeräten (Gamepad)	Reale Gegenstände bzw. Spieler selbst werden zum Spielgerät.
Interaktion mit anderen erfolgt durch den Computer	Interaktion erfolgt auch auf natürliche Art und Weise
Spiel konzentriert sich auf ein Medium	Integration unterschiedlicher Medien

Tabelle 2-1: Unterschiede zwischen klassischen Computerspielen und Pervasive Games [Prinz 2006]

2.3 Frameworks

Soweit nicht anders angegeben basiert dieses Kapitel auf [Chen 2004].

Es definiert den Begriff Framework in Abschnitt 2.3.1 und stellt dessen Aufgaben, dessen Position im Schichtenmodell einer Anwendung in Abschnitt 2.3.2 und dessen unterschiedliche Ausprägungen in Abschnitt 2.3.3 dar.

2.3.1 Definition

Frameworks sind wieder verwertbare Anwendungsbausteine, die einen allgemeinen Rahmen für jeweils spezifische konkrete Anwendungen zur Verfügung stellen [LahRay 2006]. Im Gegensatz zu Klassenbibliotheken, die von Entwicklern in ihre Projekte eingebunden werden und so Teil der Applikation sind, ist der Ansatz bei Frameworks umgekehrt. Frameworks werden durch den Entwickler instanziiert und die eigenen Klassen werden durch Vererbung selbst Teil des Frameworks. Dabei wird häufig der Kontrollfluss umgekehrt (*engl*. Inversion of Control), was dann oft als Hollywood-Prinzip bezeichnet wird: „Don't call the framework, the framework calls you" [LahRay 2006].

In dem Standardwerk Entwurfsmuster der Gang of four[13] zur objektorientierten Programmierung definieren die Autoren ein Framework als

> „eine Menge kooperierender Klassen, welche die Elemente eines wieder verwendbaren Entwurfs für eine bestimmte Art von Software darstellen. Ein Framework bietet eine Architekturhilfe beim Aufteilen des Entwurfs in abstrakte Klassen und beim Definieren ihrer Zuständigkeiten und Interaktionen. Ein Entwickler passt das Framework für eine bestimmte Anwendung an, indem er Unterklassen der Frameworkklassen bildet und ihre Objekte zusammensetzt" [GamHelJohVli 2007].

2.3.2 Schichtenmodell einer Framework-Anwendung

Wie in Abbildung 2-5 ersichtlich liegt im Schichtenmodell einer Framework-Anwendung das Framework zwischen der Business-Logik und dem zugrunde liegenden Foundation-Framework der benutzen Programmiersprache. Im Folgenden werden die oberen Schichten detaillierter erläutert.

[13] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides

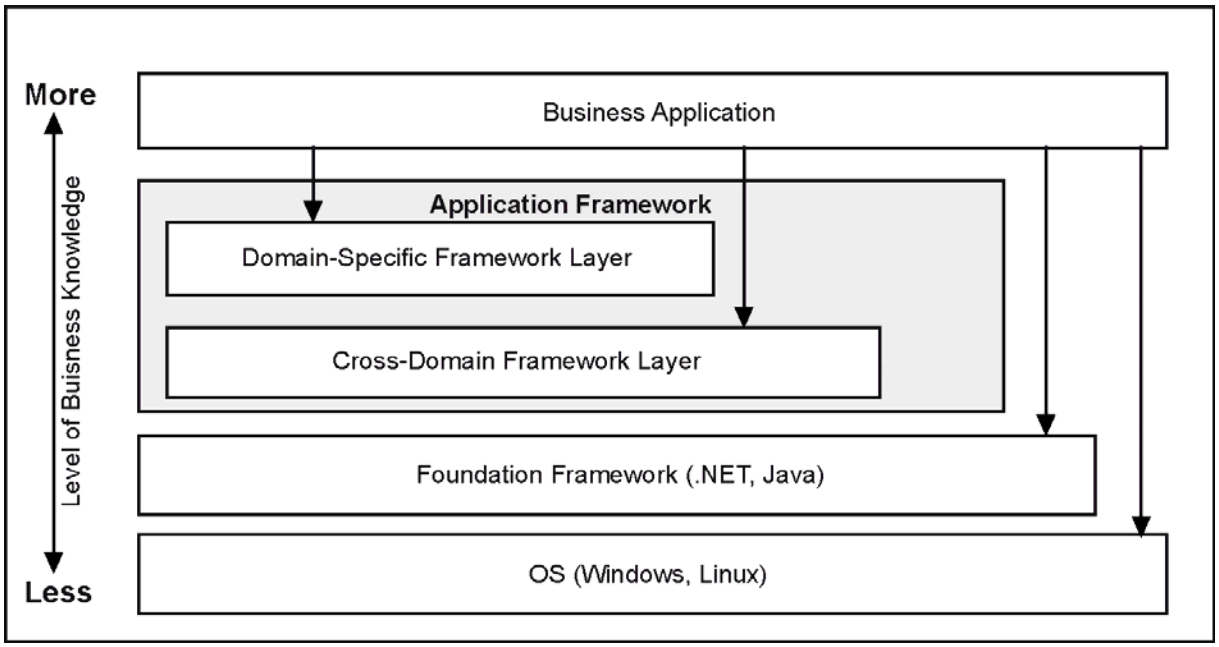

Abbildung 2-5: Schichten in einer Anwendung, die auf einem Framework aufbaut [Chen 2004]

Business-Application

Die oberste Schicht stellt die Logik der Anwendung dar, für die die Entwickler verantwortlich sind. Hier implementieren die Entwickler die individuell für die Anwendung geltenden Funktionen. Änderungen an der Logik der Anwendung werden im Regelfall nur hier durchgeführt.

Application Framework

Domain-Specific Framework Layer

In dieser Schicht stellt das Framework Anwendungs-Logik zur Verfügung. Diese ist relevant für die Lösung von Aufgabenstellungen aus bestimmten Anwendungsbereichen, z.B. für e-Commerce. Frameworks, die hauptsächlich oder ausschließlich aus dieser Schicht bestehen, werden als vertikal bezeichnet.

Cross-Domain Framework Layer

In dieser Schicht wird die Implementierung von Lösungen zu bestimmten Technologien realisiert, z.B. für Web Services. Frameworks, die hauptsächlich oder ausschließlich aus dieser Schicht bestehen, werden als horizontal bezeichnet.

2.3.3 White- und Black-Box-Frameworks

Je nach Aufbau und Struktur werden Frameworks als White-Box-Frameworks und Black-Box-Frameworks bezeichnet. Die beiden Eigenschaften schließen sich nicht aus. Solche hybriden Frameworks werden als Grey-Box-Frameworks bezeichnet.

2.3.3.1 White-Box-Frameworks

Die von Gamma u. a. angesprochene Vererbung ist eine Eigenschaft von White-Box-Frameworks [GamHelJohVli 2007]. Das Framework besteht größtenteils aus abstrakten Oberklassen, die der Anwendungsentwickler durch konkrete Implementierungen in eigenen Unterklassen nutzt. Diese Art des Frameworks ist einfacher zu entwerfen und zu entwickeln, gilt jedoch als unflexibler, da der Kontrollfluss fest mit dem Framework verankert ist und hat den Nachteil, dass für die Nutzung des Frameworks eine genauere Kenntnis der Implementierungs-Details der Framework-Klassen von Nöten ist.

2.3.3.2 Black-Box-Frameworks

Im Unterschied zu White-Box-Frameworks besteht ein Black-Box-Framework aus fertigen konkreten Klassen, die durch Komposition an Schnittstellen miteinander verbunden werden.
Aufgrund der verwendeten Komposition, sind Black-Box-Frameworks flexibler. Entwickler können verschiedene Komponenten auswählen und in unzähligen Kombinationen miteinander verbinden. Die interne Implementierung der Klassen bleibt unsichtbar, so dass der Anwendungsentwickler keine Einblicke in diese hat, diese aber auch nicht benötigt. Es ist nur Kenntnis über die Schnittstellen, die public-Methoden und die Einstellungen notwendig.
Die Entwicklung eines Black-Box-Frameworks ist aufwendiger und schwieriger im Entwurf als die eines White-Box-Frameworks, da das Framework flexibel auf möglichst viele denkbare Anwendungsszenarien anwendbar sein muss.

2.4 *GPS*

Dieses Kapitel gibt einen kurzen Einblick in die *GPS*-Technologie (offiziell NAVSTAR *GPS* für *engl.* Navigational Satellite Timing and Ranging – Global Positioning System) zur Ortsbestimmung.

Abschnitt 2.4.1 führt zunächst kurz in die verwendete Technik ein, in Abschnitt 2.4.2 wird auf die erreichbare Genauigkeit von *GPS* in Mobiltelefonen eingegangen.

2.4.1 Technik

Die als *GPS* (Global Positioning System) bezeichnete Technik dient der Lokalisierung des *GPS*-Empfängers. Mit Hilfe von 24 bis 32 die Erde umkreisenden Satelliten, von denen zu mindestens vier Satelliten zeitgleich Funkkontakt bestehen muss, kann eine 3-dimensionale Position auf der Erde ermittelt werden. Diese besteht aus Breitengrad (latitude), Längengrad (longitude) und Höhe über NN (altitude). Neben den Ortsdaten enthält das Signal noch den Zeitpunkt der Messung. Diese Art der Ortsbestimmung ist zurzeit weltweit flächendeckend, auf dem Land und auf dem Meer, möglich. Begrenzt ist der Empfang jedoch in Gebäuden, da dort im Regelfall kein optischer Kontakt zu den Satelliten möglich ist.

2.4.2 Genauigkeit

Mit herkömmlichen in Mobiltelefonen oder anderen einfachen *GPS*-Geräten verbauten Empfängern sind Genauigkeiten von unter 10 Metern möglich. Dies variiert jedoch stark nach Höhe, Wetterlage und Empfänger. Mit Hilfe besonderer Techniken und Geräte ist eine deutliche Steigerung der Genauigkeit möglich, da diese jedoch heutzutage in Mobiltelefonen noch nicht zur Verfügung stehen, ist jene für diese Arbeit nicht relevant.

2.5 *Android*

Dieses Kapitel beruht sofern nicht anders angegeben auf [BecPan 2009] und [Google 2009].

Dieses Kapitel gibt einen allgemeinen Überblick über das mobile Betriebssystem *Android*, dessen Ursprung, Architektur und die grundlegende Arbeitsweise des *Android* Development Kit bis zur Entwicklung eigener *Android*-Software mit einem Emulator.

2.5.1 Allgemeines

Im November 2007 gab die Firma *Google*™ gemeinsam mit der dafür gegründeten *Open Handset Alliance*™[14] (der u. a. *T-Mobile*, *Telecom Italia*, *Motorola*, *Samsung* und *Intel*®angehören) die Entwicklung eines Betriebssystems für Smartphones bekannt. Unter dem Namen *Android* wurde eine auf einem *Linux*™-Kernel arbeitende Entwicklungsplattform geschaffen, mit *Java*™ als Implementierungssprache für Anwendungen [Stäuble 2008]. Die Version 1.0 der Software erschien im September 2008 [Morrill 2008]. Mit dem *T-Mobile* G1 (*HTC* Corporation) erschien in den USA erst einen Monat später das erste Handy mit *Android* als Betriebssystem [Lüders 2008].
Im Dezember 2009 ist die aktuelle Version des SDK 2.0.1 erschienen. Bis zum Ende des Jahres 2009 erwartet *Google*™ auf dem Markt 18 verschiedenen *Android*-Handys von mindestens acht unterschiedlichen Herstellern [Richtel 2009].

2.5.2 Architektur

Basis der *Android*-Architektur (siehe Abbildung 2-6) ist der *Linux*™ Kernel 2.6, welcher als Betriebssystemgrundlage dient. Den Kern der Laufzeitumgebung bildet die Dalvik Virtual Machine (DVM). Jede gestartete *Android*-Anwendung startet in einer eigenen DVM, dies hat neben großen Sicherheitsvorteilen zur Folge, dass ein sterbender Prozess nur die eigene Anwendung abstürzen lässt.
Programmiert werden *Android*-Anwendungen in *Java*™. Zur Verfügung steht ein Großteil der API der *Java*™ Standard Edition (J2SE). Der normale *Java*™-Compiler javac übersetzt den Quellcode in gewohnten *Java*™-Bytecode. Die Umwandlung in den für die DVM lesbaren dex-Code übernimmt das mit dem *Android* Development Kid ausgelieferte dx-Tool. Die *.dex-Dateien zusammen ergeben dann die auf den *Android*-Systemen ausführbaren *.apk-Dateien.

Die in C/C++ entwickelten Standardbibliotheken von *Android* stellen vielfältige Funktionen für Web- und Datenbankzugriffe, Multimedia-Ansteuerungen und 2D- und 3D-Grafik zur Verfügung.

[14] Eine vollständige Liste aller der Open Handset Alliance angehörigen Firmen und weitere Informationen finden sich unter http://www.openhandsetalliance.com.

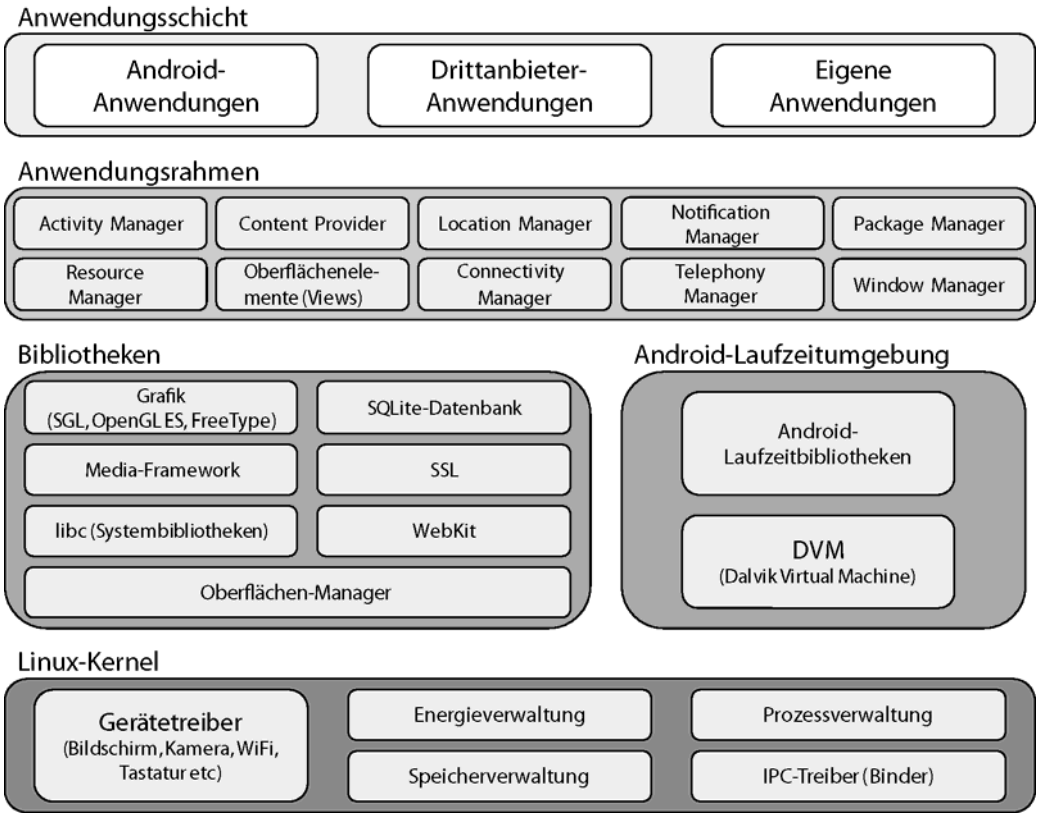

Abbildung 2-6: Die *Android*-Systemarchitektur [BecPan 2009]

Der Anwendungsrahmen kapselt die zugrunde liegende Hardware von der Anwendungsschicht und stellt spezifische Funktionen zur Verfügung, die im Folgenden, soweit für diese Arbeit relevant, erklärt werden.

Activity Manager

Activities sind die Basis jeder Interaktion einer Anwendung mit dem Nutzer. Als Faustregel gilt, dass jede Oberfläche einer Anwendung einer Activity zuzuordnen ist. Der zugehörige Manager überwacht u. a. deren Lebenszyklus: Eine durch eine andere Activity verdeckte Activity wird z.B. pausiert bis sie wieder im Vordergrund erscheint oder ganz beendet wird.

Location Manager

Dieser Manager bildet die Schnittstelle zu Technologien, welche den aktuellen Standort als Geopunkt liefern können. Im Regelfall ist dies ein eingebautes *GPS*-Modul im Handy, jedoch gibt es auch Möglichkeiten der Positionsermittlung über Netzwerk oder mit per *Bluetooth®* mit dem Handy verbundener Empfänger.

Notification Manager

Über den Notification Manager kann der Nutzer des Handys auf Systemebene, ohne die aktuelle Anwendung zu unterbrechen, über Ereignisse wie z.B. eingehende SMS oder Terminhinweise aus dem Kalender informiert werden. Dies kann über leuchtende oder

blinkende LEDs am Gerät, ein Piktogramm in der Statuszeile, Sounds oder Vibrationen geschehen.

Resource Manager

Als Ressourcen werden in *Android* alle externen (nicht Quellcode-Dateien) bezeichnet. Dies können u. a. verschiedene Grafik- oder XML-Dateien sein. Zur Lokalisierung einer Anwendung können z.B. alle Texte einer Sprache in eine andere XML-Datei in einem sprachspezifischen Ordner (z.B. für Deutschland `res/values-de/strings.xml`) geschrieben werden. Die Verknüpfung aus dem Quellcode zu dieser XML-Datei stellt der Resource Manager her.

Oberflächenelemente (Views)

Die auf Activities platzierten Oberflächenkomponenten (*engl*. auch Widgets) wie Schalter, Auswahllisten, Tabellen etc. zur Oberflächengestaltung werden in *Android* als Views bezeichnet.

Connectivity Manager

Der Connectivity Manager steuert die Netzwerkverbindungen. So wird sowohl auf geeignetere Netze gewechselt, z.B. beim Betreten eines Gebäudes von *UMTS*™ auf WLAN, als auch das System auf Betriebssystemebene über Änderungen wie z.B. den Abbruch einer Verbindung informiert.

2.5.3 Entwicklung

Die Entwicklung einer *Android*-Anwendung ist auch ohne ein passendes Endgerät möglich. *Google*™ liefert mit dem SDK (Software Development Kit) das *Android* Virtual Device (AVD) mit. Es handelt sich hierbei um ein Tool zur individuellen Konfiguration eines Emulators (siehe Abbildung 2-7), der mit Ausnahme echter Telefonanrufe alle Funktionen der *Android*-Handys unterstützt. Die Konfiguration erlaubt z.B. die Auswahl verschiedener Handy-Typen und Auflösungen sowie die Bestimmung der Netzwerkgeschwindigkeit und der Netzwerklatenz

Abbildung 2-7: Die Startoberfläche eines *Android*-Emulators [Android 2009]

Mit Hilfe des Dalvik Debug Monitor Servers (DDMS) können im laufenden Emulator nicht nur Geodaten und Telefonanrufe simuliert werden, sondern auch Informationen zu laufenden Threads und dem aktuell verwendeten Heap eingesehen werden.

2.5.4 *Google Maps™*

Google™ liefert die Handys mit direktem Zugriff auf die *Google*™ Dienstleistung *Google Maps*™ aus und bietet damit Entwicklern unzählige Möglichkeiten zur Entwicklung ortsbasierter Software. Zur Nutzung der Maps API ist eine Registrierung bei *Google*™ inkl. der Generierung eines API-Schlüssels notwendig.

Google™ bietet mit der Klasse `android.location.Geocoder` sowohl die Zuordnung von einem gegebenen Geopunkt zur nächstgelegenen Adresse, also auch die Ermittlung eines Geopunktes zu einer gegebenen Adresse. Nicht möglich ist jedoch die Ermittlung, ob sich eine gegebene *GPS*-Koordinate auf einer Straße, in einem Haus oder auch in Bergen, auf Wasser oder im Wald befindet. Dies ist nur für den Nutzer auf der darstellbaren Karte ersichtlich, eine technische Abfrage dazu gibt es nicht.[15]

[15] Das Paket `com.google.googlenav` zur Entwicklung eigener Echtzeitnavigation und Streckenführung ist von Google aus rechtlichen Gründen mit Erscheinen des SDK 1.1 entfernt worden.

2.6 Technische Umsetzung von Online-Rollenspielen

In diesem Kapitel werden die technischen Grundlagen, auf denen Online-Rollenspiele, basieren vorgestellt. In Abschnitt 2.6.1 wird zunächst die typische Architektur eines Online-Rollenspiels vorgestellt. Die Abschnitte 2.6.2 und 2.6.3 widmen sich den verwendeten Netzwerkprotokollen und der Manipulation in Online-Rollenspielen.

2.6.1 Architektur

Große Online-Rollenspiele wie *World of Warcraft*™, *Everquest® 2* und *Anarchy Online*™ verwenden als Basisarchitektur alle eine Client-Server-Architektur. Die Spielwelt wird in einzelne Zonen geteilt, die jede für eine begrenzte Anzahl Spieler (im Beispiel *Everquest® 2* für 200 [FriRitSch 2005]) ausgelegt ist. Jede dieser Zonen wird durch einen oder mehrere einzelne Server betreut (siehe Abbildung 2-8). Ein World-Server koordiniert die Anfragen der Clients. Spezielle Login-Server verteilen die Berechtigungen nach erfolgreicher Authentifizierung der Spieler. Die Daten der Spielwelten werden in relationalen Datenbanken [ZhaKemDen 2008] gespeichert.

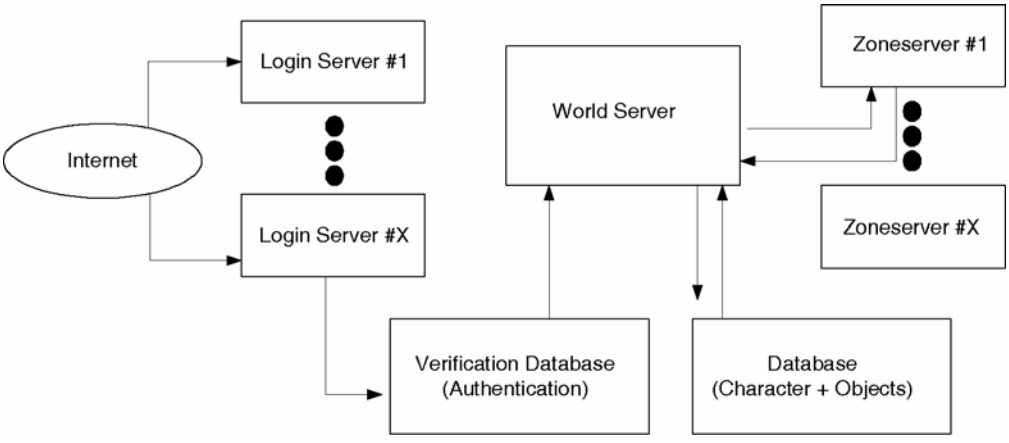

Abbildung 2-8: Beispiel einer Multi-Server-Architektur eines Online Rollenspiels [FriRitSch 2005]

Nach Rieche u. a. haben Server in Online-Spielen folgende Aufgaben:

- Kenntnis über alle Spielerpositionen haben und den Clients die Position anderer Spieler in der Nähe mitteilen
- Kontrolle aller Monster, NPCs, Gegenstände etc., allgemein über alle dynamischen Aspekte des Spiels
- Kampfberechnung

- Chatverwaltung
- Verwaltung der Account- und der Charakterdaten der Spieler [RieWehFouNiePetCar 2007]

Der Client hat in einem MMORPG also nur die Aufgabe, die Verbindung zum Server zu halten, dessen übermittelte Daten zu visualisieren und die Eingaben der Spieler an den Server zu senden. Eine Ausnahme bildet die Bewegung der Charaktere. Zur Sicherung eines flüssigen Spielablaufes werden vom Spieler ausgeführte Bewegungen in den meisten Spielen jedoch zusätzlich vom Client auf Korrektheit geprüft und dann visualisiert. Sollte die nach der Übermittlung der Bewegung an den Server stattfindende Prüfung ein anderes Ergebnis als auf Clientseite liefern, wird die Visualisierung entsprechend korrigiert.

2.6.1.1 Alternative Peer-to-Peer

Es gibt einige wissenschaftliche Studien zur Senkung der Serverlasten von Online-Spielen durch die Verwendung von Peer-to-Peer-Technologien. Laut Assiotis u. a. sind diese Ansätze aber für kommerzielle Spiele nicht zielorientiert, da der Entwickler keine Kontrolle über den Client hat [AssTza 2006]. Auch Manipulation des Clients kann nicht ausgeschlossen werden (siehe 2.6.3). Laut Mönch u. a. kann durch so eine Manipulation des Clients der betrügende Spieler Zugriff auf Informationen haben, die ihm eigentlich nicht zustehen, da sein Rechner ja in der zeitgleichen Funktion als Client und Server auch Informationen für andere Mitspieler kennt [MönGriMid 2006].

2.6.2 Netzwerkprotokoll

Als Netzwerkprotokoll werden für die Übertragungen des Spielzustandes fast ausnahmslos TCP basierende Point-to-Point Verbindungen eingesetzt. [GriHal 2006] halten TCP trotz des erhöhten Datenaufkommens nicht zwangsweise für langsamer als UDP und gehen davon aus, dass TCP aufgrund der verwendeten Firewall-Technologien auch weiterhin im MMORPG-Sektor Standard bleibt. Generell sind Netzwerkverzögerungen in Online-Rollenspielen aber laut Rieche u. a. weniger relevant wie z.B. in Online-Taktik-Shootern wie *Counter Strike*™[16], da die Kämpfe hauptsächlich von Taktik und Glück abhängen und nicht von Geschicklichkeit [RieWehFouNiePetCar 2007].

[16] Entwickler: Valve, 2000 – Das Spiel handelt von Gefechten zwischen Terroristen und einer Antiterroreinheit.

UDP wird in einigen Spielen für spezielle Dienste eingesetzt, Blizzard zum Beispiel verwendet für *World of Warcraft*™ UDP für die Übertragung des spielinternen Voice-Chats[17].

2.6.3 Cheating

Die Manipulation in Computer-Spielen wird allgemein als Cheating (engl. für Betrug) bezeichnet. Diese ist in Online-Rollenspielen ein großes Problem [MönGriMid 2006]. Unehrliche Spieler können den Spielclient und die Netzwerkkommunikation manipulieren oder den Speicher des Systems auslesen und so an zusätzliche Informationen kommen, die ihnen nicht zustehen. Die Gefahr der Manipulierbarkeit der Clients steigt, wenn diese mehr Aufgaben als nur die Darstellung vom Server übermittelter Daten haben.

Um das Problem manipulierter Clients zu lösen, verwenden *World of Warcraft*™ und *Everquest® 2* eine Protected-Server-Technologie. Die Server senden den Clients nur verschlüsselte Daten, die diese nur lesen können, wenn sie nicht manipuliert wurden.
Das Erhalten zusätzlicher Informationen wird generell durch die verwendete Architektur verhindert. Die Clients übernehmen nur darstellende und keine berechnenden Aufgaben und erhalten vom Server immer nur die Informationen, die sie aktuell für die Darstellung benötigen.

[17] World of Warcraft bietet eine Funktion an, sich mit Hilfe eines Mikrofons mit anderen Mitspielern per Sprache zu verständigen.

3. Analyse

Dieser Abschnitt spezifiziert die Funktionen und Eigenschaften des im Rahmen dieser Arbeit zu entwickelnden mobilen Rollenspiels. Dafür werden im ersten Abschnitt 3.1 die Aspekte eines Rollenspiels im Allgemeinen und die der Pervasive Games in einen gemeinsamen Kontext zusammengeführt und die daraus resultierenden Probleme erläutert. Im Anschluss daran werden in Abschnitt 3.2 anhand der Rollenspiel-Grundlagen aus Kapitel 2.1 die Funktionen und Grenzen des Frameworks herausgearbeitet. Der Abschnitt 3.3 vergleicht die Bewegung des Charakters in einem realen und virtuellen Kontext und erörtert deren Vor- und Nachteile. Zwei bereits bestehende Rollenspiele mit Pervasive Gaming-Elementen werden in Abschnitt 3.4 vorgestellt und mit den Zielen dieser Arbeit verglichen.

3.1 Rollenspiele und Pervasive Games

Dieses Kapitel soll die Rollenspiele allgemein und die Pervasive Games in einem Kontext vereinen, um aufzuzeigen wie ein derartiges mobiles Rollenspiel am Ende aussehen könnte.
In Abschnitt 3.1.1 erfolgt die Bildung eines gemeinsamen Kontextes für beide Aspekte und damit eine grundlegende Vorstellung des Spielkonzepts welches mit dem Framework abgedeckt werden soll. Das anschließende Kapitel 3.1.2 zeigt einige für die weitere Ausarbeitung relevante Konsequenzen aus der Zusammenführung auf.

3.1.1 Bildung eines gemeinsamen Kontextes

Das Merkmal der Pervasive Games ist, wie in Kapitel 2.2 vorgestellt, die Vermischung aus virtueller und physischer Realität, möglichst unter Einbeziehung einer Ortsabhängigkeit.

Für den Begriff mobile Rollenspiele werden die beiden Realitäten in der Form vereint, dass der Spieler selbst seinen Charakter im Rollenspiel physisch repräsentiert. Im Unterschied zum Live-Rollenspiel passiert dies aber je nach Wunsch zu jedem beliebigen Zeitpunkt, auch in seinem Alltagsleben. Das Spiel muss dafür nur auf dem Handy aktiv sein. Ist dies der Fall,

bewegt sich die Spielfigur anhand der übermittelten *GPS*-Daten durch die Welt und befindet sich stets an der Position, an der sich auch der Spieler befindet.

Alle die spielrelevanten Informationen werden auf dem *Android*-Handy dargestellt inkl. einer Karte, welche die aktuelle Position des eigenen Charakters sowie die der Mitspieler und ggf. zum Spiel gehöriger Örtlichkeiten darstellt.

Die in Kapitel 2.2 beschriebenen Eigenschaften der Pervasive Games werden somit in folgender Form übertragen:

- Das Spiel wird auf einem mobilen Endgerät gespielt und ist daher ortsunabhängig.
- Die Position des Charakters in der Spielwelt ist abhängig von der realen Position des Spielers, damit ist Ortssensitivität gewährleistet.

Daraus ergeben sich folgende Konsequenzen:

- Da das Spiel immer auf dem Handy angeschaltet sein kann, ist es jederzeit gegenwärtig und verfügbar.
- Der Spieler selber ist die Spielfigur.
- Der Spieler kann das Spielen mit alltäglichen Aufgaben wie Einkaufen o. Ä. verbinden.
- Interaktion kann direkt ohne technische Hilfsmittel durch Sprache und Gesten erfolgen.

Von den von Prinz [Prinz 2006] herausgearbeiteten und in Tabelle 2-1 ersichtlichen Eigenschaften ist die kooperative Spielform bereits durch die zugrunde liegende Basis der Online-Rollenspiele gegeben. Für die in dieser Arbeit herausgearbeitete Spielform „mobiles Rollenspiel“ ist die Integration unterschiedlicher Medien nicht umsetzbar. Außerdem sind die in das Spiel integrierten realen Gegenstände, die zum Spielgerät werden sollen, nicht realisierbar (Begründung siehe 3.1.2).

3.1.2 Konsequenzen der Zusammenführung

Aus der Zusammenführung der beiden Aspekte ergeben sich Konsequenzen für die Entwicklung mobiler Rollenspiele.

3.1.2.1 Sinneswahrnehmung

In den Pen & Paper-Rollensspielen spielt sich die komplette Sinneswahrnehmung des Charakters ausschließlich in der Fantasie der Spieler ab. Gelegentlich eingesetzte Zeichnungen oder Miniaturen dienen im Regelfall nur der Vermeidung von Missverständnissen. In Computer- und Online-Rollenspielen entfällt nun ein Großteil der Fantasie, gerade bei gestiegener Grafikfähigkeit, in Bezug auf die visuelle und auditive Wahrnehmung. In einem Live-Rollenspiel gilt das Gleiche. Der Charakter sieht was der Spieler wahrnimmt. Da im Live-Rollenspiel alle Spieler verkleidet sind und die Szenarien den Örtlichkeiten angepasst werden, wird das Spielerlebnis im Regelfall wenig gestört.
In einem mobilen Rollenspiel sieht und hört der Spieler also das Gleiche wie sein Charakter, ähnlich wie im Live-Rollenspiel. Der Unterschied besteht nur darin, dass er vieles wahrnimmt was nicht zur Spielwelt gehört (z.B. andere Personen), Objekte anders sieht als sie in der Spielwelt eigentlich sind (anstatt einer mittelalterlichen Gasse bewegt er sich zwischen Hochhäusern) und auch nur wirklich vorhandene Dinge sehen kann. Alles Weitere muss er sich in seiner Fantasie vorstellen.
Diese Wahrnehmung einer anderen Realität hat Einfluss auf die Handlungsmöglichkeiten des Charakters.

3.1.2.2 Handlungsmöglichkeiten

Im Live-Rollenspiel sind die Handlungsmöglichkeiten ähnlich wenig eingeschränkt wie im Pen & Paper-Rollenspiel. Grenzen sind nur physikalischer und logischer Natur. Am Computer ist man jedoch immer an die programmierten Funktionen gebunden. Wenn ein Entwickler z.B. im Spiel nicht vorgesehen hat, dass man einen Stuhl hochheben kann, dann ist das im Spiel auch nicht möglich. Selbst wenn der Stuhl visuell in dem Spiel dargestellt wird.
Sieht man einen Stuhl nun jedoch im mobilen Rollenspiel (z.B. vor einem Straßencafe), kann man ihn physikalisch anheben. Die Physik lässt das zu, die Logik auch. Man hat ähnliche Freiheiten in der Handlung wie im Pen & Paper-Rollenspiel oder Live-Rollenspiel. Der das Spiel überwachende Computer kann jedoch weder wissen, dass dort ein Stuhl steht, noch überprüfen, ob dieser Stuhl angehoben wurde und wann und wo er wieder abgestellt wurde. Es fehlt eine Kontrollinstanz.

3.1.2.3 Kontrollinstanz

Rollenspiele haben wie jede Art von Spielen Regeln. Für die Überprüfung der Einhaltung dieser sind im Pen & Paper-Rollenspiel und im Live-Rollenspiel die Spielleiter zuständig, am Computer übernimmt das eben dieser.
Im mobilen Rollenspiel sollte das so auch vom Computer übernommen werden. Ein Computer kann jedoch immer nur die Dinge auswerten, für die ihm Daten zur Verfügung stehen. Diese erhält er z.B. über Sensoren, hier über den *GPS*-Empfänger für die Positionsbestimmung oder als Eingabe über das Handy. Da der Stuhl keinen Sensor hat, könnte man dem Spieler die Möglichkeit geben, eben diese Eingabe („Ich habe einen Stuhl hochgehoben und 2 m weiter in Richtung Norden wieder abgestellt“) dem System mitzuteilen. Ganz davon abgesehen, dass dies nicht praktikabel ist, da es dann unendlich viele Eingabemöglichkeiten für jede beliebige Handlung geben müsste, fehlt die Kontrollinstanz, die prüft, ob das wirklich passiert ist und der Spieler dies nicht einfach nur eingegeben hat.

3.1.2.4 Zusammenfassung

Tabelle 3-1 vergleicht die in der Arbeit behandelten Arten von Rollenspielen hinsichtlich Sinneswahrnehmung, Handlungsmöglichkeiten und Kontrollinstanz. Dort wird die Einzigartigkeit der Form der Sinneswahrnehmung in mobilen Rollenspielen ersichtlich. Die Handlungsmöglichkeiten hingegen vermischen die Freiheit der Pen & Paper- und der Live-Rollenspiele mit den aus der Software gegebenen Grenzen der digitalen Spiele. Die zuständige Kontrollinstanz kann jedoch nur einen Teil der Handlungsmöglichkeiten überwachen. Die sich daraus ergebenen Konsequenzen sind im kommenden Kapitel 3.1.2.5 aufgeführt.

	Sinneswahrnehmung des Charakters	Handlungsmöglichkeiten des Charakters	Kontrollinstanz zur Regeleinhaltung
Pen & Paper-Rollenspiele	In der Fantasie der Spieler	Nur durch die physikalischen und logischen Gesetze der Spielwelt begrenzt	Spielleiter
Computer-Rollenspiele	Visuelle und auditive Wahrnehmung über Bild- und Soundausgabe des Computers	Begrenzt im Rahmen der programmierten Möglichkeiten	Computer
Online-Rollenspiele	Visuelle und auditive Wahrnehmung über Bild- und Soundausgabe des Computers	Begrenzt im Rahmen der programmierten Möglichkeiten	Computer
Live-Rollenspiel	Sicht des Charakters mit wenigen Störfaktoren	Begrenzt durch die Spiellogik und die auf der Erde geltenden physikalischen Gesetze	Spielleiter
Mobiles Rollenspiel	Aus der Sicht der Spieler – aber nicht unbedingt dem Szenario der Welt entsprechend	In der Wirkung begrenzt durch die programmierten Möglichkeiten, nicht aber in der Ausführung	Nur der Computer – der hat jedoch nicht auf alle Informationen Zugriff.

Tabelle 3-1: Sinneswahrnehmung, Handlungsmöglichkeiten und Kontrollinstanz im Vergleich

3.1.2.5 Konsequenz für Mobile Rollenspiele

Aufgrund fehlender Sensoren für die reale Welt kennt das bei mobilen Rollenspielen als Kontrollinstanz fungierende Computersystem nur zwei Dinge: Die vom System selbst generierten Spielinformationen (Charakterwerte, ggf. Monster, virtuelle Gegenstände etc.) und die Position aller Mitspieler. Es gilt allgemein, dass es nur Funktionen im Spiel geben darf, welche die Kontrollinstanz auch prüfen kann.

Das schließt Folgendes aus:

- Handlungsmöglichkeiten mit unbeteiligten nicht an dem Spiel teilnehmenden Personen
- Handlungsmöglichkeiten mit real existierenden Objekten
- Spielrelevante Auswirkungen auf Charaktere, die nicht vom Computersystem überprüfbar sind

3.2 Eigenschaften eines mobilen Rollenspiels

Dieses Kapitel arbeitet die Eigenschaften und Funktionen, die mobile Rollenspiele haben können, heraus.
Dazu werden in Abschnitt 3.2.1 alle Eigenschaften von Pen & Paper-, Live-, Computer- und Online-Rollenspielen auf ihre Übertragbarkeit auf mobile Rollenspiele unter Berücksichtigung der Eigenschaften von Pervasive Games geprüft und deren Konsequenzen für die Funktionen des Frameworks erörtert.

3.2.1 Übertragbarkeit bestehender Eigenschaften

Dieses Kapitel prüft die in dem Grundlagenkapitel 2.1 vorgestellten Eigenschaften von Rollenspielen auf ihre technische und logische Übertragbarkeit auf ein mobiles Rollenspiel. Alle hier zur Veranschaulichung benutzen Fertigkeiten, Gegenstände und Eigenschaften sind geläufige Beispiele aus unzähligen Rollenspielen aller Art.

3.2.1.1 Spielfigur

In Computer- und Online-Rollenspielen wird der vom Spieler gespielte und im Sinne eines Rollenspiels dargestellte Charakter durch einen Avatar[18] vertreten. Die Besonderheit bei dem virtuelle und physische Realität vermischenden Ansatz der Pervasive Games ist, dass bei einem mobilen Rollenspiel dieser Art nun der Körper des Spielers selbst seine Spielfigur, sein Avatar, ist.

Konsequenzen für das Framework
Da ein Spieler in der realen Welt immer nur eine Figur zur gleichen Zeit darstellen kann, ist das aus einigen Computer-Rollenspielen bekannte Spielkonzept, dass ein Spieler eine Gruppe mehrerer Charaktere steuert, nun vollkommen ausgeschlossen.
Folglich muss das Framework die Möglichkeit bieten, für jeden Teilnehmer des Spiels einen Charakter spielen zu können, dessen Positions-Daten per *GPS* zu ermitteln und die Charakterdaten zu verwalten (siehe 3.2.1.2).

[18] hier: virtuelle Repräsentation einer Person, grafischer Stellvertreter

3.2.1.2 Charakterentwicklung

Die Kapitel 2.1.4.2 und 2.1.5.2 machen die breit gefächerten Möglichkeiten zur Abbildung und Verbesserung der Eigenschaften und Fertigkeiten von Charakteren in Computer- und Online-Rollenspielen ersichtlich.

Charakterentwicklung → Bestandteile

Für die Bestandteile der Charakterentwicklung „Klassen und/oder Rassenwahl“ und „Charakter-Eigenschaftswerte“ gibt es keine technischen Gründe, die gegen eine Umsetzungen dieser Art in einem mobilen Rollenspiel sprechen. All diese Dinge sind in nicht-mobilen Rollenspielen durch Zahlen oder Symbole repräsentiert und somit auch in einem mobilen Spiel auf die gleiche Art und Weise darstellbar.

Der Designer eines mobilen Rollenspieles muss jedoch für diese Punkte bei der Entwicklung einige Entscheidungen zur Spiellogik treffen. Ein Problem kann z.B. bei der Implementierung der Option „Rassenwahl“ auftreten. Jede beliebige Person, unabhängig von der eigenen Körpergröße, kann einen Zwergen spielen und muss diesen auch in der Öffentlichkeit gegenüber Mitspielern darstellen.

In Pen & Paper-Rollenspielen, in denen auch jeder Spieler unabhängig von seinen eigenen körperlichen Grenzen darstellt, was er möchte, geschieht dies im Gegensatz zum mobilen Ansatz dort ausschließlich in der Fantasie. Es gibt keinen Avatar, keine Spielfigur.

Da bei mobilen Rollenspielen aber der Spieler identisch mit der Spielfigur ist, stellt sich die Frage, in wie weit hier Charakterdaten auf den Spieler zutreffen sollten. Dies sollte jeder Entwickler individuell entscheiden. Ein ähnliches Problem tritt auch bei Live-Rollenspielen auf, dort sind die Spieler jedoch verkleidet und es gilt im Regelfall, dass jeder nur darstellen darf, was ihm auch möglich ist. Eine 1,90m große Person wird also keinen Zwergen spielen.

Charakterentwicklung → Bestandteile → Fertigkeiten

Bei den speziellen Charakter-Fertigkeiten treten ähnliche Probleme auf. Nehmen wir an, ein mobiles Rollenspiel beinhaltet die Möglichkeit zu schleichen, um unbemerkt ein Monster passieren zu können. Nun kann wie in Computer- oder Online-Rollenspielen der Erfolg dieses Unterfangens vom Zufall abhängig gemacht werden. Je nach Spielsystem ist der Charakter erfolgreicher, desto besser er schleichen kann. Der Spieler wählt nun die Fertigkeit auf seinem Mobiltelefon aus und versucht, sich an dem Monster vorbei zu schleichen. Allerdings gibt es eigentlich keinen Bedarf für das Auswählen der Fertigkeit, da das System nicht überprüfen

kann, wie sich die Person fortbewegt hat. Selbst eine *GPS*-gestützte Ermittlung der Bewegungsgeschwindigkeit scheitert an der Ungenauigkeit der *GPS*-Empfänger bei so niedrigen Geschwindigkeiten. Die in vielen Rollenspielen so spannende Fertigkeit des Schleichens verkommt also zu einem aktivieren und deaktivieren auf dem Handy.
Auch für fast alle gängigen Zauber wie z.B. Heilzauber, Verwandlungszauber oder das Werfen von Feuerbällen gilt das gleiche Problem. All diese magischen Fertigkeiten verlaufen vollkommen effektfrei und sind durch andere Spieler auch nur dann zu erkennen, wenn entweder ihr eigenes *Android*-Handy sie mit einer Nachricht darauf hinweist („Der Spieler vor dir hat sich gerade in einen Feuer speienden Drachen verwandelt.“) oder der Mitspieler das selbst äußert („Ich bin jetzt ein Drache und spucke gerade Feuer!“).

Charakterentwicklung → Verbesserung des Charakters

Die Punkte „Lernen durch Stufenanstieg“ und „Lernen durch Anwendung“ lassen sich sowohl technisch als auch von der Spiellogik her problemlos auf ein mobiles Rollenspiel übertragen. Für „Lernen durch Trainer“ gilt das auch, mit der Voraussetzung, dass es auch Nicht-Spieler-Charaktere (siehe 3.2.1.3) in der Spielwelt gibt, denn Trainer werden im Regelfall Nicht-Spieler-Charaktere sein.

Der Punkt „Verbessern durch das Finden von Gegenständen“ hingegen ist etwas diffiziler. Das ein Spieler z.B. aufgrund eines Schwertes mehr Schaden beim Kämpfen macht, durch einen Geschicklichkeitsring besser ausweichen kann und daher weniger Schaden bekommt oder durch ein Amulett mehr magische Energie zum Zaubern hat, ist noch gut umsetzbar. Das sind alles Gegenstände, die nur Zahlenwerte verändern und für die Mitspieler von geringer Bedeutung sind (für die ist nur wichtig, wie gut jemand kämpfen kann, nicht primär warum).
Schwieriger wird es jedoch mit Gegenständen die Fertigkeiten (z.B. „Schuhe des Schleichens“ siehe oben) geben oder sogar Gegenstände, die die realen körperlichen Fertigkeiten verändern würden. Ein Beispiel dafür wären Schuhe, mit denen man schneller laufen kann. Nur weil der Computer die Information verarbeitet, dass der Charakter nun schneller laufen kann, tut das der jeweilige Spieler aber noch lange nicht.

Konsequenzen für das Framework

Die konkrete Umsetzung der Charakterentwicklung in Computer- und Online-Rollenspielen unterscheidet sich von Spiel zu Spiel sehr stark. Im Gegensatz dazu, muss der Entwickler bei mobilen Rollenspielen wie oben an Beispielen gezeigt, sehr individuell abwägen, was für sein

Spielkonzept an Kompromissen in der Spiellogik akzeptabel ist. Generell gilt, dass Schwierigkeiten immer dann auftreten, wenn mindestens einer der folgenden Punkte oder einer der Aspekte unter 0 zutrifft.

- Spielrelevante Auswirkungen, die nicht visuell darstellbar sind (z.B. die Möglichkeit zu fliegen)
- Spielrelevante Auswirkungen, die nur wirken, wenn der Spieler sie physisch umsetzt (z.B. Schuhe zum schnelleren Laufen) oder ein Lähmungszauber, der für langsamere Bewegungen sorgt

Da es jedoch wie bereits beschrieben keine Argumente gegen eine technische Realisierung der angesprochenen Aspekte gibt, muss das Framework für individuelle Charakterdaten und deren Entwicklung eine Schnittstelle bieten. Wie die im Detail aussieht, muss der jeweilige das Framework nutzende Entwickler entscheiden.

Um dem Spieler die Möglichkeit zu geben individuelle Charaktere mit unterschiedlichen Eigenschaften zu spielen, benötigt das Framework eine Funktion zur Charaktergenerierung. Diese wird nur bei der Erschaffung eines neuen Charakters aufgerufen.

3.2.1.3 Spielwelt

Szenario

Der Wahl des Szenarios sind im Prinzip keinerlei Grenzen gesetzt, so lange bei der Fortbewegung keine physikalischen Grenzen überschritten werden. Ein Rollenspiel in einer Welt, in der alle Charaktere fliegen können, ist nicht umsetzbar. Die typischen Szenarien Fantasy, Horror, Science-Fiction und Endzeit führen aber zu keinen derartigen Problemen. Der Entwickler des Rollenspiels muss sich nur immer vor Augen führen, dass die dargestellte Welt, die der Spieler visuell und auditiv wahrnimmt, die reale Welt des 21. Jahrhunderts ist. Eine Darstellung des Szenarios kann nur durch Grafiken und Texte auf dem Handy geschehen.

Örtlichkeit

Die Umsetzung der für Rollenspiele typischen Örtlichkeiten gestaltet sich bei Gebäuden als äußerst schwierig. Aufgrund der dort nicht funktionierenden *GPS*-Technologie ist eine Umsetzung jeder Art von Bewegung, die nicht unter freiem Himmel stattfindet,

ausgeschlossen. Die in fast allen Rollenspielen verwendeten Dungeons sind nicht umsetzbar. In realen Kellern oder Höhlen würde das *GPS*-Modul keine Signale empfangen und virtuelle Dungeons würden eine Sub-Karte der normalen *Google Maps*™-Karte verlangen. Auf diese Problematik wird in Kapitel 3.3 noch genauer eingegangen und eine mögliche Lösung vorgestellt.

Für das Betreten realer Gebäude gilt das Problem des nicht funktionierenden *GPS*-Empfängers wie oben erläutert wurde. Eine praktikable Möglichkeit der Implementierung von Gebäuden gibt es bei virtuellen, real nicht existierenden Gebäuden: Sobald der Spieler sich nahe des Geopunktes des Gebäudes befindet, hat er über die Spielsoftware auf dem Mobiltelefon die Möglichkeit dies zu betreten. Wählt der Spieler diese Funktion aus, kann danach ein Auswahlmenü angezeigt werden, welches die Optionen in dem Gebäude anbietet. Handelt es sich um einen Händler, kann hier z.B. ein- oder verkauft werden. Auf dem gleichen Weg kann das Gebäude auch wieder verlassen werden. Was nicht möglich ist, ist ein wirkliches Bewegen durch verändern der eigenen Position in dem Gebäude.
Wie das System damit umgeht, wenn ein Spieler während er sich virtuell in einem Gebäude befindet, fortbewegt, muss der Entwickler für sein Spiel entscheiden. Denkbar sind folgende Varianten:

1.) Das Gebäude wird automatisch verlassen und alle in dem Gebäudemenü gerade durchgeführten Aktionen (z.B. Handel) abgebrochen.
2.) Der Spieler kann die Aktionen beenden, bis er eigenständig durch Auswahl des Menüpunktes das Gebäude verlässt und wird dann an die neue Position gesetzt, an der er sich laut *GPS*-Empfänger jetzt befindet. Der Charakter macht somit einen Sprung in der Welt.

Die Positionierung von Gebäuden ist ebenfalls schwierig. Wie bereits in Kapitel 2.5.4 gibt es für *Google Maps*™ unter *Android* keinerlei Möglichkeit für einen gegebenen Geopunkt, bestehend aus Längen- und Breitengrad zu ermitteln, was sich an dieser Position befindet. Eine zufällige Positionsbestimmung kann daher nicht sicherstellen, dass sich dieser Ort nicht auf Wasser oder anderen nicht erreichbaren Gebieten wie militärischen Sperrgebieten befinden würden. Eine feste Positionierung solcher Gebäude setzt dies jedoch für alle Bereiche voraus, in denen das Spiel gespielt werden soll.

Sichtbereich

Es ist zwar mit der *GPS*-Technologie möglich, die Richtung einer Bewegung, z.B. bei einem fahrenden Auto zu ermitteln. Dies ist bei sich sehr unregelmäßig bewegenden Fußgängern jedoch aufgrund der Unschärfe der Messung nicht möglich. Die Blickrichtung eines Spielers kann damit also nicht erfasst werden. Sinnvoll erscheint, immer anzunehmen, dass ein Spieler einen Rundumblick hat. Alle spielrelevanten Informationen (andere Spieler, eventuelle virtuelle Gebäude etc.), die sich in einem gewissen Umkreis um die Position des Spielers befinden, müssen dargestellt werden. Wie groß dieser Umkreis ist, kann der Entwickler des Spiels selbst bestimmen.

Zusätzlich nicht möglich ist die Beschränkung des Sichtbereiches durch real existierende Hindernisse wie Gebäude. Da es keine Möglichkeit gibt, mit der von *Google*™ für *Android* zur Verfügung gestellten Technologie ein 3-dimensionales Hindernis zu erkennen (siehe 2.5.4), heißt das konkret: Ein Spieler der direkt auf die Hauswand eines Gebäudes blickt, kann auf seinem Handy sehen, ob sich hinter dem Gebäude spielrelevante Informationen befinden. Selbst wenn diese Erkennung für Gebäude möglich wäre, würden die Sicht versperrende mobile Hindernisse wie LKWs oder andere Fahrzeuge weiterhin nicht erkannt werden.

Monster

Aus dem gleichen Grund wie die zufällige Verteilung der Gebäude schwierig ist (siehe 3.2.1.3 „Örtlichkeit“), kann auch nicht sichergestellt werden, dass die Routen zufällig bewegender Monster nicht durch Gebäude oder anderes eigentlich nicht passierbares Gebiet führen. Die Alternative, feste Routen für Monster zu geben, erscheint auf den ersten Blick nahe liegend. Wenn ein mobiles Rollenspiel weltweit spielbar sein soll, ist die dann anfallende Arbeit nicht mehr verhältnismäßig, da solche Routen für jeden beliebigen Ort der Erde erzeugt werden müssten.

Nicht-Spieler-Charaktere

Die Implementierung von Nicht-Spieler-Charakteren ist in der gleichen Form möglich wie bei Computer- und Online-Rollenspielen. Sollten diese in Gebäuden stehen gelten die oben unter „Örtlichkeit“ gemachten Einschränkungen. Für Reisende Nicht-Spieler-Charaktere gelten die unter „Monster“ gemachten Einschränkungen.

Persistenz

Die Persistenz-Eigenschaft wirkt sich im mobilen Rollenspiel genau so aus wie in Online Rollenspielen.

Instanzen

Die in aktuelleren Online-Rollenspielen - besonders auch beim Marktführer *World of Warcraft*™ - eingesetzten Instanzen spezieller Gebiete, ist auf mobile Rollenspiele nicht zu übertragen.

Auf der einen Seite gibt es ähnliche Probleme wie schon bezüglich des Betretens von Gebäuden erläutert. Auf der anderen Seite würden damit Grundgedanken der Pervasive Games verletzt werden, da sich auf die Art und Weise ggf. mehrere Spieler real am gleichen Ort befinden, im Spiel aber keinen Kontakt haben können, da sie sich in unterschiedlichen Instanzen befinden.

Handelssystem

Ein Handelssystem kann in mobilen Rollenspielen analog zu denen in Online-Rollenspielen implementiert werden. Da die Reisezeiten in der realen Welt um ein Vielfaches länger sind als in denen von Online-Rollenspielen (in *World of Warcraft*™ dauert eine Reise zu Fuß von einem zum anderen Ende der Welt unter einer Stunde), könnten sich in mobilen Rollenspielen sogar deutlich unterschiedliche Preisgefüge einpendeln. In Online-Rollenspielen kosten die gleichen Waren im Regelfall überall in der Spielwelt dasselbe.

Konsequenzen für das Framework

Da es möglich, aber nicht notwendig ist, dass es in einem mobilen Rollenspiel virtuelle Gebäude gibt, sollte das Framework eine Schnittstelle für die Implementierung jener bieten. Diese muss die Funktionen „betreten“ und „verlassen“ implementieren. Für die konkreten individuellen Funktionen der Gebäude ist der Entwickler des mobilen Rollenspiels zuständig.

Das Framework muss in einem durch den Entwickler konfigurierbaren Radius um die aktuelle Position des Spielers herum alle spielrelevanten Elemente darstellen.

Da es für die Routen, auf denen sich Monster bewegen mehrere Möglichkeiten gibt (zufällig, fest codiert, eine Mischung aus beidem…), genügt es, wenn das Framework in der Lage ist, Monster darzustellen. Für deren Bewegung ist der Entwickler des mobilen Rollenspiels zuständig.

Für Nicht-Spieler-Charaktere gilt Ähnliches wie für Gebäude. Die zu implementierenden Funktionen heißen dann allerdings „ansprechen“ und „Gespräch beenden“. Für den eigentlichen Dialog und deren spielrelevanten Funktionen ist der Entwickler zuständig.

Für die Einhaltung der Persistenz ist es notwendig, dass beim Starten der Software auf dem *Android*-Handy alle spielrelevanten Daten dem Spiel neu mitgeteilt werden. Dies hat zur Folge, dass es keinen Sinn macht, spielrelevante Daten auf dem Mobiltelefon zu speichern, da diese sich in der persistenten Welt immer ändern und speziell auch dann, wenn der Nutzer das Spiel gerade nicht aktiv hat. Außerdem muss das Spiel mit einer Login-Funktion zur Anmeldung eines Charakters am System versehen sein, um sich als Nutzer zu identifizieren und beim erneuten Starten des Programms den eigenen Charakter weiterspielen zu können.

Für die Implementierung eines Handelssystems werden Nicht-Spieler-Charaktere oder Gebäude benötigt, um mit der Software handeln zu können. Eine konkrete Implementierung für einen derartigen Handel ist also nicht notwendig.
Für den Handel mit einem Mitspieler ist jedoch eine Schnittstelle notwendig, die den Spielern ermöglicht Gegenstände und/oder Geld zu handeln.

3.2.1.4 Spielziel

Endgegner und Quests
Die zu Endgegner und Quests gemachten Aussagen in Kapitel 2.1.5.2 gelten für mobile Rollenspiele in der gleichen Form.

Konsequenzen für das Framework
Das Framework muss eine Schnittstelle zur Visualisierung und Überprüfung von Quests anbieten.

3.2.1.5 Kampfsystem

Für das Kampfsystem kommt wie in Online-Rollenspielen nur die Variante des Echtzeitkampfes in Frage, da die persistente Spielwelt nicht pausiert werden kann. Ein Problem, welches bei Online-Rollenspielen nicht auftritt, ist die potentiell sehr schnelle Bewegung des Spielers während des Kampfes. Dort kann ein Charakter zwar vor Gegnern

fliehen, dies jedoch nur in der normalen Geschwindigkeit. Das Monster kann dann dem Charakter folgen und versuchen, ihm im Wegrennen Schaden zuzufügen. Dies endet entweder durch den Tod des Spielercharakters, den Tod des Monsters (z.B. dank zu Hilfe gekommener anderer Spieler oder Nicht-Spieler-Charaktere) oder den Abbruch der Verfolgung durch die künstliche Intelligenz des Monsters.
Bei der Übernahme dieses Konzepts auf mobile Rollenspiele treten jedoch Probleme mit der Bewegungsgeschwindigkeit der Spieler auf. Macht man die Flucht eines Charakters abhängig von der Bewegung des Spielers, können körperlich fitte Spieler besser fliehen. Außerdem könnten Charaktere, deren Spieler sich während eines Kampfes bewegen, den Kampfbereich mit dem Monster ungewollt verlassen und damit den Kampf beenden. Dies kann z.B. an Bahnhöfen auftreten, wenn ein Spieler im Zug sitzt, welcher während eines Kampfes anfährt. Es bieten sich dafür z.B. folgende Möglichkeiten an, mit diesem Problem umzugehen:

- Sobald ein Spieler in einen Kampf verwickelt ist, kann der Gegner an die Position des Spielers gebunden sein. Das Monster bewegt sich also während des Kampfes mit, unabhängig von der Geschwindigkeit des Gegners. Eine Flucht ist nur durch eine explizite Auswahl der Funktion möglich. Andere Spieler können dann in den Kampf nicht eingreifen und es ist auch kein zeitgleicher Kampf mehrerer Spieler gegen einen Gegner möglich. Diese Lösung enthält weniger Eigenschaften des Pervasive Gaming, da eine Flucht nicht mehr durch die Bewegung des Spielers möglich ist und auch die potentielle Interaktion mit Mitspielern entfällt.
- Wenn das Problem akzeptiert werden soll, findet die Flucht eines Spielers immer dann statt, wenn dieser einen bestimmten Bereich um den Standort des Monsters verlässt. Dies hat zur Konsequenz, dass die körperlichen Möglichkeiten des Spielers und nicht Zahlenwerte oder Auswahlmöglichkeiten auf dem Bildschirm für eine erfolgreiche Flucht relevant sind. Dies benachteiligt jedoch körperlich schwächere Spieler oder Spieler die in Regionen spielen, in denen eine schnelle Fortbewegung nicht möglich ist.

Kampf gegen andere Spieler

Für den Kampf gegen andere Spieler gelten auf der einen Seite die gleichen Aussagen wie für Online-Rollenspiele, auf der anderen Seite ist ein Kampf zwischen Spielern sehr nahe an den Grundsätzen der Pervasive Games. Im Gegensatz zu einem Kampf gegen Monster oder Nicht-Spieler-Charaktere, die ja nur virtuell wären, wären andere Mitspieler reale Gegner.

Konsequenzen für das Framework

Das Framework muss eine Schnittstelle für die Kampfberechnung anbieten. Unabhängig davon ob virtuelle Gegner oder Mitspieler bekämpft werden, erfolgt die Beurteilung des Ausganges des Kampfes nur auf Berechnungen anhand der Charakterdaten der Kontrahenten. Die Entscheidung des Umgangs mit der Bewegung des Spielers im Kampf muss der Entwickler tragen. Daher muss die Schnittstelle der Kampfberechnung so flexibel konzipiert werden, dass hier eine individuelle Lösung für jedes mobile Rollenspiel entwickelt werden kann.

3.2.1.6 Grafik

Die Fragestellung nach der grafischen Darstellung entfällt für mobile Rollenspiele. Die Sicht des Charakters ist die des Spielers. Dieser sieht jedoch die reale und nicht die virtuelle Welt in dem Spiel. Was der Charakter anstelle der realen Welt sehen würde ist in Texten und Bildern auf dem Mobiltelefon angezeigt, größtenteils mit kleinen Bildern auf der Karte. Dies gedanklich zu übertragen auf die wirkliche Sicht des Spielers, ist die Aufgabe seiner Fantasie.

3.2.2 Praktische Einschränkungen

Dieses Kapitel erwähnt zwei praktische Einschränkungen, die für die Nutzung des beschriebenen Spielkonzeptes mobiles Rollenspiel auftreten. Der Unterabschnitt 3.2.2.1 behandelt Sicherheitsbedenken im Straßenverkehr, während der Abschnitt 3.2.2.2 auf die Kosten für mobiles Internet eingeht.

3.2.2.1 Sicherheit im Straßenverkehr

Ein Aspekt, der bei der Entwicklung eines mobilen Rollenspiels zwar keine technische Konsequenz hat, aber dennoch bei der Umsetzung dieses Spielkonzeptes im Hinterkopf behalten werden sollte, ist die Frage nach der Sicherheit im Straßenverkehr. Sollte ein Spieler z.B. vor einem Monster oder Charakter fliehen wollen, um den Tod seines eigenen Charakters zu verhindern, könnte dieser den einzigen Fluchtweg in der Überquerung einer Straße sehen. Da das Spielsystem in solchen Situationen keine Rücksicht auf Ampelschaltungen nehmen kann und der Spieler sich in Zeitdruck befindet, entsteht dabei, auch für unbeteiligte Personen, eine erhöhte Unfallgefahr.

3.2.2.2 Internetgebühren

Entwickler und Spieler heutiger mobiler Rollenspiele müssen mit dem gleichen Problem umgehen, wie ihre Vorreiter, die vor über 10 Jahren die Online-Rollenspiele entwickelt und gespielt haben. Die Kosten für dauerhafte mobile Internetverbindungen sind heute im Verhältnis ähnlich zu denen der damaligen Verbindungskosten für stationäres Internet. Die Kosten für mobile Flatrates sinken zwar stetig, haben aber noch lange nicht die Größenordnung für normale DSL-Flatrates erreicht. Solange dies nicht der Fall ist, wird die potentielle Spielerzahl für derartige Spiele immer weit unter der für Online-Rollenspiele liegen.

3.3 Realer oder virtueller Kontext

Dieses Kapitel stellt neben dem bisher als Grundlage für die Analyse verwendeten realen Kontext eine Alternative mit einem fiktiven virtuellen Kontext vor.
Abschnitt 3.3.1 beschreibt kurz den realen Kontext, Abschnitt 3.3.2 geht dann ausführlich auf die Eigenschaften, Vor- und Nachteile der Wahl eines virtuellen Kontexts ein. Das Kapitel 3.3.3 stellt grafisch den Ablauf eines Ortswechsels des Spielers für beide Kontexte dar.

3.3.1 Realer Kontext

Beim realem Kontext wird die reale Karte (auf *Android*-Systemen die *Google Maps*™-Karte) zu der Karte, auf der sich der Spielercharakter entsprechend der Bewegungen des Spielers bewegt. Dieser für Pervasive Games typische Ansatz sorgt für eine optische und gedankliche Verknüpfung der beiden Welten. Wenn der Spieler das Spiel verlässt oder die *GPS*-Verbindung aussetzt und keine Positionsbestimmung möglich ist, wird der Spieler bei der nächsten Herstellung der Verbindung das Spiel an der neuen realen Position fortsetzen.
Die Eigenschaften, Möglichkeiten und Beschränkungen des realen Kontextes entsprechen den in den vorherigen Kapiteln durchgeführten Analysen. Einen Vergleich zu denen des virtuellen Kontexts zeigt Tabelle 3-2.

3.3.2 Virtueller Kontext

Im virtuellen Kontext wird über die reale Karte eine fiktive für das Spiel individuelle virtuelle Karte gelegt. Eine Bewegung in der realen Welt wird auf der fiktiven Karte im virtuellen Kontext durchgeführt. Es wird also eine eins-zu-eins Abbildung der Bewegung durchgeführt. Eine Bewegung in der realen Welt um eine Bogensekunde[19] auf einem Breitengrad (ca. 31 m) entspricht dann eben dieser Entfernung auf der virtuellen Karte. Für die muss nur der Maßstab der Karte definiert sein. Sollte in dieser Variante der Spieler das Spiel verlassen oder die *GPS*-Verbindung aussetzen, wird bei Wiederherstellung der Verbindung die Position des Charakters sich weiterhin an der alten Stelle befinden. Ein Ortswechsel ist also nur mit *GPS* möglich. Dafür muss die letzte Position des Charakters immer zwischengespeichert werden.

Eigenschaft	Realer Kontext	Virtueller Kontext
Echte Verschmelzung der virtuellen Spielwelt und der realen Welt	+	-
Reale direkte Kommunikation mit Sprache oder Gesten ohne technische Hilfsmittel	+	-
Im Spiel integrierte virtuelle Orte befinden sich immer zu jedem Zeitpunkt am selben realen Ort – die Konsistenz ist von Spielsitzung zu Spielsitzung gegeben	+	-
Alle Bereiche der Spielwelt sind betretbar und nicht durch reale Hindernisse wie Flüsse oder Meere blockiert.	+	-
Sichtbegrenzung durch Hindernisse möglich	-	+
Keine Probleme bei weltweiten ggf. zufälligen Verteilungen von Gebäuden	-	+
Monstern und sich bewegenden Nicht-Spieler-Charakteren können zufällige Bewegungsrouten zugewiesen werden	-	+
Implementierungsmöglichkeit der für Rollenspiele typischen Dungeons und anderen Sub-Karten	-	+
Modifikation der Bewegungsgeschwindigkeit durch Gegenstände, Zauber oder andere Effekte möglich	-	+
Beständigkeit der Position in der Spielwelt trotz Verlust von *GPS*-Empfang oder Verlassen des Spiels während der eigenen Bewegung	-	+
Besser in spezifische Rollenspielszenarien passende Karte	-	+
Unterschiedliche Karten je nach Spiel	-	+
Gemeinsames Spiel mit räumlich entfernten Spielern möglich	-	+
Besserer Datenschutz (Visualisierung der Position des Spielers)	-	+

Tabelle 3-2: Vor- und Nachteile von realem und virtuellem Kontext

[19] 1 Grad = 60 Bogenminuten = 360 Bogensekunden

3.3.2.1 Vorteile des virtuellen Kontextes

Viele im Laufe der Analyse (Kapitel 3.2) herausgearbeiteten Nachteile werden mit dem System der virtuellen Karte aufgehoben. Da die Karte nun nur noch eine begrenzte Größe hat, können Hindernisse im Programmcode definiert werden. Somit ist es möglich, zu verhindern, dass Monster oder Nicht-Spieler-Charaktere durch Gebäude laufen und es tritt das Problem der Gebäudeverteilung nicht mehr auf. Aus dem gleichen Grund ist es nun auch möglich, den Sichtbereich von Charakteren durch fiktive 3D-Hindernisse zu begrenzen. Auch das Einführen von Sub-Karten wie Dungeons, eines der zentralen Elemente vieler Rollenspiele, ist damit möglich. Charaktere können, wenn sie sich auf der virtuellen Karte am Eingang eines Dungeons befinden, über ein Auswahlmenü diesen betreten. Das System blendet auf die zugehörige Karte um und ab dem Moment werden alle Bewegungen des Spielers in der realen Welt auf die Bewegung des Charakters in dem Dungeon abgebildet.
Entwickler, denen die individuellen Szenarien sehr wichtig sind, profitieren durch virtuelle Karten, die sie sowohl optisch als auch geographisch beliebig gestalten können. Ebenso ist es nicht zu unterschätzen, dass ein Spieler sich in einem virtuellen Kontext anonym bewegt. Es sind für andere Spieler keine Rückschlüsse aus der virtuellen Spielposition auf die reale Position des Spielers möglich. Dies gewährt besseren Datenschutz.

3.3.2.2 Nachteile des virtuellen Kontextes

Ein Problem kann auftreten, wenn auf der virtuellen Karte sich z.B. nach Westen freies Land befindet, um in die Richtung weitergehen zu können, dies aber nicht möglich ist, da in der realen Welt dort Häuser oder sogar Wasser oder ein Meer die Fortbewegung behindert. An dieser Stelle besteht für den Spieler die einzige Möglichkeit darin, das Spiel zu verlassen, sich dann weiter nach Osten zu bewegen (oder im Norden oder Süden wieder Raum für eine weitere Reise nach Westen zu haben), um dann wieder in das Spiel einzuloggen. Da sich die virtuelle Position nicht verändert hat, kann der Spieler nun durch die jetzt mögliche Bewegung nach Westen auch im Spiel nach Westen reisen.
Ein weiteres Problem ist, dass dieses Konzept weniger Eigenschaften der Pervasive Games umsetzt. Der Spieler nutzt die reale Welt nun nur noch für die Fortbewegung. Einen Bezug zum realen Kontext gibt es darüber hinaus nicht mehr.
Auch auf eine direkte Kommunikation müssen die Spieler verzichten, da sie sich in den seltensten Fällen zufälligerweise auch zeitgleich am selben realen Ort befinden, wenn sie sich am selben virtuellen Ort befinden. Dies kann durch ein Chatsystem kompensiert werden.

Etwas gewöhnungsbedürftig ist die Tatsache, dass ein Gebäude, welches sich gestern noch an einer bestimmten realen Straßenecke befand, heute nach Aus- und wieder Einloggen, woanders befindet, wenn das Mobiltelefon während des abgeschalteten Zustandes an einen anderen Ort bewegt wurde.

3.3.3 Eine Bewegung im Vergleich

Abbildung 3-1 zeigt zur Veranschaulichung einen Ablauf von denkbaren Bewegungen im realen und virtuellen Kontext.

3.3.4 Konsequenz für das Framework

Die beiden Ansätze des realen und virtuellen Kontexts haben ihre Berechtigung, da sie vollkommen unterschiedliche Vor- und Nachteile haben. Jeder Entwickler eines Spiels sollte diese gegeneinander abwägen können. Daher soll das Framework auf der einen Seite für Rollenspiele, die eine stärkere Ausrichtung zu den Pervasive Games haben, ermöglichen, einen realen Kontext zu verwenden. Auf der anderen Seite kann für mobile Rollenspiele, in denen die klassische Spielbarkeit von Rollenspielen im Vordergrund steht, der virtuelle Kontext verwendet werden.

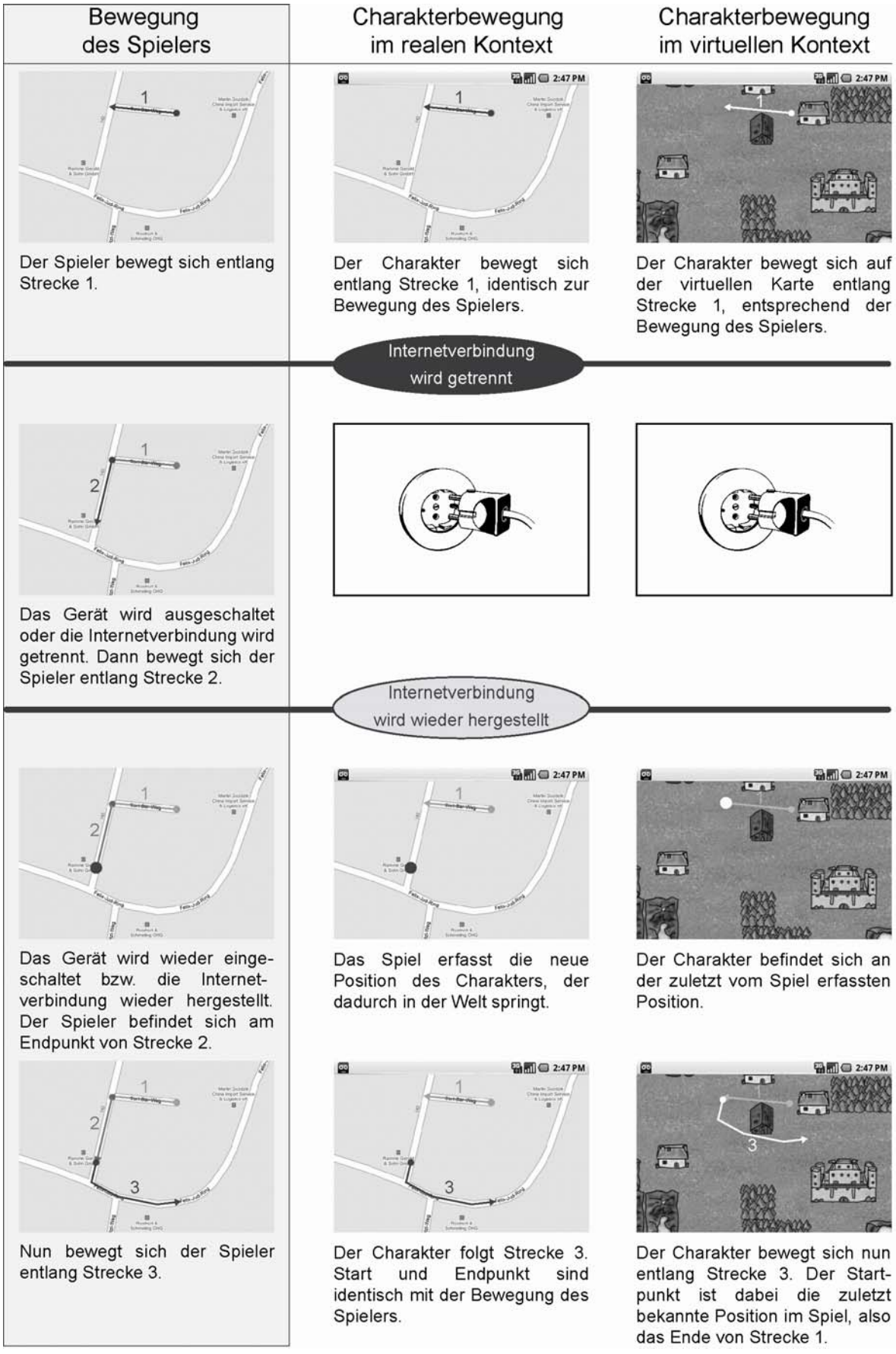

Abbildung 3-1: Realer und virtueller Kontext: Bewegung im Vergleich

3.4 Bestehende Spielkonzepte

In diesem Kapitel werden in den Abschnitten 3.4.1 und 3.4.2 zwei bereits bestehende Spiele für *Android* mit Elementen aus Rollenspielen und Pervasive Games vorgestellt und in Bezug auf die Analysen dieser Arbeit eingestuft.

3.4.1 Relativia

Die Informationen in diesem Kapitel beruhen auf [Polyclefsoftware 2009], [James 2009a] und [Janus 2007b].

Das Spiel Relativia vermischt klassische Rollenspielelemente mit Elementen aus Puzzlespielen[20]. Entwickelt wurde es für die 2. Auflage der *Android* Developer Challenge[21], konnte dort jedoch keinen Preis gewinnen. Es steht ein kostenloser Download über den *Android* Market[22] zur Verfügung.

3.4.1.1 Spielablauf

Der Spieler generiert zu Beginn seinen Charakter aus je vier unterschiedlichen Rassen und Klassen. Durch die eigene Bewegung in der realen Welt kann der Spieler nun zu auf der Karte angezeigten Dungeons und Märkten gehen. Der Markt dient dem Kauf von Ausrüstung für den Charakter, die Dungeons hingegen öffnen ein Puzzle-Duell gegen einen softwaregesteuerten Gegner. Dieses Puzzle erinnert in der Grundidee an das Strategiespiel „Vier gewinnt“, bindet jedoch die Charakterwerte und sich in Besitz befindenden Gegenstände des Charakters in das Spiel ein. Der Spieler muss in diesem Puzzle durch geschicktes Nutzen seiner Fertigkeiten die Lebenspunkte des Gegners auf null senken, bevor seine eigenen Lebenspunkte verbraucht sind. Screenshots des Spiels zeigt Abbildung 3-2.

[20] Gemeint sind hier nicht die klassischen Puzzle in denen ein Bild zusammengesetzt werden muss, sondern die meist abstrakten Denkspiele für den Computer (Beispiele siehe http://www.spiele-zone.de/seiten/puzzle.php).
[21] Google hat zur Förderung der Marktverbreitung von Android bisher zwei Android Developer Challenges veranstaltet, in denen Android-Software prämiert wurde. Der Gewinner der zweiten Ausgabe erhielt 250.000$.
[22] Der „Android Market“ ist ein von Google betriebenes Portal zur Vermarktung der für Android entwickelten Software.

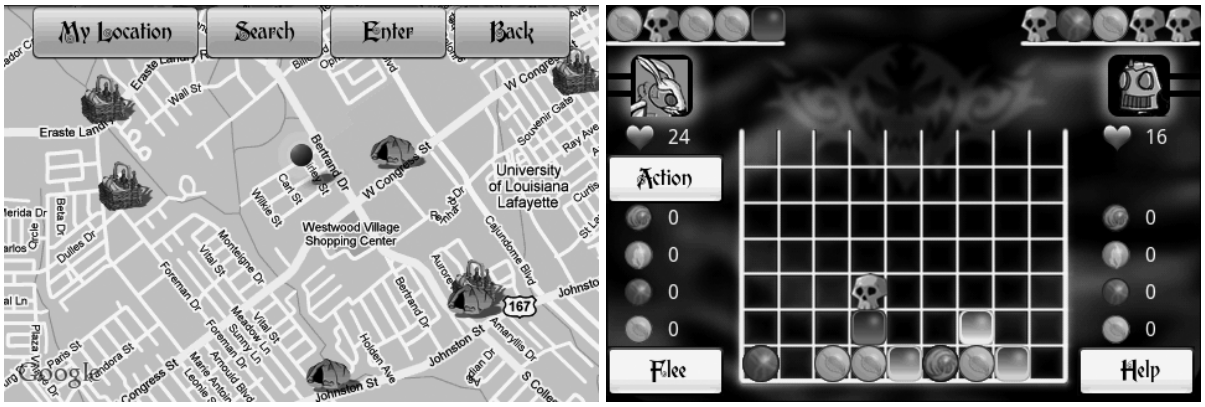

Abbildung 3-2: Relativia - links: Karte mit Spielelementen, rechts: Puzzle-Kampf [Polyclefsoftware 2009]

3.4.1.2 Rollenspiel-Elemente

Relativia beinhaltet folgende Rollenspielelemente:

- Charaktergenerierung mit Rassen- und Klassenauswahl
- Individuelle Fertigkeiten und Zauber je nach Rasse und Klasse
- Finden von Geld durch das Besiegen von Gegnern
- Verbesserung des Charakters durch Erwerben von Gegenständen

3.4.1.3 Aspekte der Pervasive Games

Folgende Elemente der Pervasive Games sind in Relativia verarbeitet:

- Das Spiel ist ortssensitiv, d.h. der Spieler muss bestimmte reale Örtlichkeiten erreichen, um den Markt oder die Dungeons nutzen zu können.
- Das Spiel ist jederzeit während des Alltagskontextes eines Spielers nutzbar. Sollte sich ein geeigneter Ort in der Nähe befinden, kann der Spieler einen Dungeon oder Markt betreten.

3.4.1.4 Technische Umsetzung

Um den in 3.2.1.3 beschriebenen Problemen für die Verteilung von Gebäuden aus dem Weg zu gehen, hat der Entwickler sich bei der Verteilung der Märkte und Dungeons für folgende Methode entschieden: Relativia durchsucht auf Knopfdruck die nahe Umgebung um die Position des Spieler mittels der Geocoder-Klasse nach Hotels und Supermärkten (verwendete Stichworte "hotel", "coffee+shop", "grocery") ab und platziert an deren *GPS*-Koordination die Märkte und Dungeons. Dieses System funktioniert in der aktuellen Relativia-Version nur für den englischen Sprachraum. Sichergestellt ist im Gegensatz zu einer zufälligen Verteilung der Örtlichkeiten jedoch, dass diese für den Spieler auch wirklich erreichbar sind.

3.4.1.5 Kritik

Relativia verzichtet auf eine zentrale Verwaltung der Spielerdaten und ist damit ein reines Einzelspielerspiel. Es gibt keinerlei Interaktionsmöglichkeiten mit anderen Spielern Es ist weder möglich, dass Puzzle gegen menschliche Spieler zu spielen, noch sind die Gegenstände mit Mitspielern handelbar. Selbst auf der Karte sind andere Spieler, die das gleiche Spiel spielen, nicht zu erkennen. Dies hat jedoch den Vorteil, dass keine permanente Internetverbindung notwendig ist, sondern diese nur zeitweise zum Laden der Kartendaten benötigt wird. Sollte die *GPS*-Verbindung abbrechen, ist ein Spiel jedoch nicht mehr möglich. Allerdings hat der Entwickler für den Fall einen Trainingsmodus des Puzzles integriert.

Damit ist Relativia genau das, was es sein möchte. Eine Umsetzung eines Puzzles mit Rollenspielelementen unter Einbeziehung der Ortssensitivität. Das Spiel genügt damit aber nicht den Anforderungen des im Rahmen dieser Arbeit zu entwickelnden Frameworks, da Relativia keinerlei Elemente eines Online-Rollenspiels beinhaltet.

3.4.2 *Parallel Kingdom*® - Age of Emergence

Parallel Kingdom® ist ein kommerzielles Rollenspiel, welches für *Android* und das *Apple iPhone*® im Jahr 2009 erschienen ist. Das Spiel ist kostenlos, jedoch stehen ausgewählte Spielinhalte nur gegen Bezahlung zur Verfügung.

3.4.2.1 Spielablauf

Der Spielablauf von *Parallel Kingdom*® kommt dem von Online-Rollenspielen sehr nahe. Man spielt ein Rollenspiel mit typischen Elementen mit verschiedenen Mitspielern. Besonders ist die von den Entwicklern gewählte Art der Fortbewegung. Für *Parallel Kingdom*® wurde eine Mischung aus dem ortssensitiven Ansatz der Pervasive Games und der klassischen Steuerung der Spielfigur über das Mobiltelefon gewählt.
Das Spiel spielt auf den realen Karten von *Google Maps*™. Der Spieler kann seine Spielfigur um einen gewissen Radius um seinen aktuellen Standpunkt mit dem Handy bewegen. Möchte er diesen Radius verlassen, ist eine reale Fortbewegung des Spielers notwendig, um den Bewegungsradius zu verschieben. Zusätzlich können mit steigender Charakterstärke und ausreichend gesammelten Rohstoffen Flaggen auf der Karte verteilt werden, die dem Charakter einen größeren Bewegungsrahmen und das Teleportieren von Flagge zu Flagge erlauben. Screenshots des Spiels zeigt Abbildung 3-3.

Abbildung 3-3: *Parallel Kingdom*® (Screenshots vom *Apple iPhone*®), links: Kampf gegen Monster auf der Straße, rechts: Kampf in einem Dungeon [PerBlue 2009]

3.4.2.2 Rollenspiel-Elemente

Parallel Kingdom® beinhaltet u. a. folgende Rollenspielelemente:

- Charakterentwicklung
- Kampf gegen Monster
- Kämpfen in Dungeons
- Eigenes Bauen von Gebäuden
- Chat-Kommunikation mit allen Mitspielern
- Kampf in Gruppen mit anderen Mitspielern
- Handel mit Mitspielern und Nicht-Spieler-Charakteren

3.4.2.3 Elemente der Pervasive Games

In *Parallel Kingdom*® ist es mit Hilfe des Flaggen-Systems nicht nur möglich, sich in der Umgebung um die aktuelle Position der Spielfigur zu bewegen, sondern über Einladung von

Freunden sogar Sprünge zu fremden Flaggen auf andere Kontinente zu machen. Eine physische Bewegung ist selten notwendig. Damit ist eine grundlegende Voraussetzung der Pervasive Games nicht erfüllt. Der Spieler bewegt sich weder in der realen Welt, um seinen Avatar zu bewegen, noch kann er direkt ohne technische Hilfsmittel mit anderen Mitspielern kommunizieren. Eine Bewegung in der realen Welt verschiebt nur den Bewegungsradius, die Spielfigur wird nicht bewegt. Auf diese Art und Weise werden auch die Dungeons realisiert. Der Spieler kann wie in herkömmlichen Spielen die Spielfigur über die Tasten des Handys zu dem Dungeon steuern, ihn virtuell betreten und mit der gleichen Steuerung sich in dem Dungeon fortbewegen.

3.4.2.4 Kritik

Parallel Kingdom® ist eines der ersten kommerziellen Rollenspiele mit Elementen der Pervasive Games für *Android*. Diese neue Spielidee wurde jedoch aus Sicht der Pervasive Games nur oberflächlich umgesetzt und nur ein Teil des neuen Elementes der Ortssensitivität mit in das Spiel integriert. Die Entwickler haben mit dem Bewegungsradius jedoch die Probleme des in 3.3 angesprochenen realen Kontextes gelöst, ohne eine virtuelle Karte zu verwenden. Es ist sowohl möglich Sub-Karten wie Dungeons zu betreten, man kann auch mit sich räumlich weit entfernt befindlichen Spielern gemeinsam spielen und der Datenschutz ist gesichert, da die aktuelle Position des Avatars nur einen sehr groben Rückschluss auf die aktuelle Position des Spielers zulässt. Nicht gelöst ist jedoch das Problem der zufälligen Monsterrouten, die durch Gebäude führen. Ein großer Vorteil durch die mangelnde Ortssensitivität ist, dass eine *GPS*-Verbindung nur notwendig ist, wenn der Bewegungsrahmen des Charakters neu gesetzt werden soll. Ansonsten funktioniert das Spiel ohne *GPS*, also auch in Gebäuden. Es ist nur eine durchgehende Internetverbindung notwendig.

Die Elemente der Online-Rollenspiele sind sehr ausführlich umgesetzt und sogar durch seltene Funktionen wie das Bauen eigener Gebäude ergänzt.

Parallel Kingdom® ist jedoch daher kein Beispiel für ein Spiel, welches mit dem in dieser Arbeit entwickelten Framework programmiert werden kann, da die Verwendung der Positionsdaten des Spielers nicht den Anforderungen an die Pervasive Games genügt.

4. Konzeption

In diesem Kapitel wird die Konzeption des Frameworks dargelegt. Nachdem in Abschnitt 4.1 auf die Gesamtarchitektur eingegangen wird, werden in Kapitel 4.2 die Anforderungen an das Framework zusammengestellt. Unter 4.3 wird ein Basissystem entworfen, was in Kapitel 4.4 hinsichtlich weiterer funktionaler Anforderungen erweitert wird. Abschnitt 4.5 stellt speziell für das hier erarbeitete Framework getroffene Entscheidungen vor.

4.1 Gesamtarchitektur

Dieses Kapitel stellt die Gesamtarchitektur der Konzeption dieses Frameworks vor. Der Abschnitt 4.1.1 geht dabei im Allgemeinen auf die verwendete Architektur ein, während der darauf folgende Abschnitt 4.1.2 Einschränkungen in der Umsetzung für diese Arbeit erläutert.

4.1.1 Allgemein

Die mobilen Rollenspiele ähneln in der Gesamtarchitektur den Online-Rollenspielen. Wie in Kapitel 2.6 beschrieben und in Abbildung 2-8 dargestellt wird dafür im Regelfall eine Client-Server-Architektur verwendet. Für das mobile Rollenspiel bedeutet dies, dass es einen Server (oder ein Netz mehrerer Server) geben muss, der alle Spieldaten speichert, Berechnungen auf deren Grundlage durchführt und die Ergebnisse an die mobilen Clients sendet. Die Clients sind nur für die Entgegennahme von Eingaben des Spielers und für die Ausgabe der vom Server übermittelten Daten zuständig.

Abbildung 4-1 zeigt eine abstrakte vereinfachte Darstellung der Minimal-Architektur eines mobilen Rollenspieles. Der Game-Server mit integriertem Login-Server zur Authentifizierung des Spielers speichert die persistenten Daten der Spielwelt in einer Datenbank. Die am Spiel teilnehmenden Mobiltelefone sind über das Internet mit dem Server verbunden.

Eine zusätzliche clientseitige Datenbank ist aus zwei Gründen nicht sinnvoll. Neben der Manipulierbarkeit der gespeicherten Daten (siehe 2.6.3) würde diese Lösung dem Anspruch der Persistenz entgegenwirken. Das lässt sich dadurch begründen, dass sich die persistente

Welt auch dann weiterentwickelt, wenn die Software auf dem Client abgeschaltet ist. Unabhängig von einer eventuellen Speicherung der Daten müssen daher bei jedem Spielstart vom Server alle für den Charakter relevanten Spieldaten neu übertragen werden. Dieser würde die in der clientseitigen Datenbank gespeicherten Daten ersetzen und daher ist eine clientseitige Speicherung in einer Datenbank nicht sinnvoll.

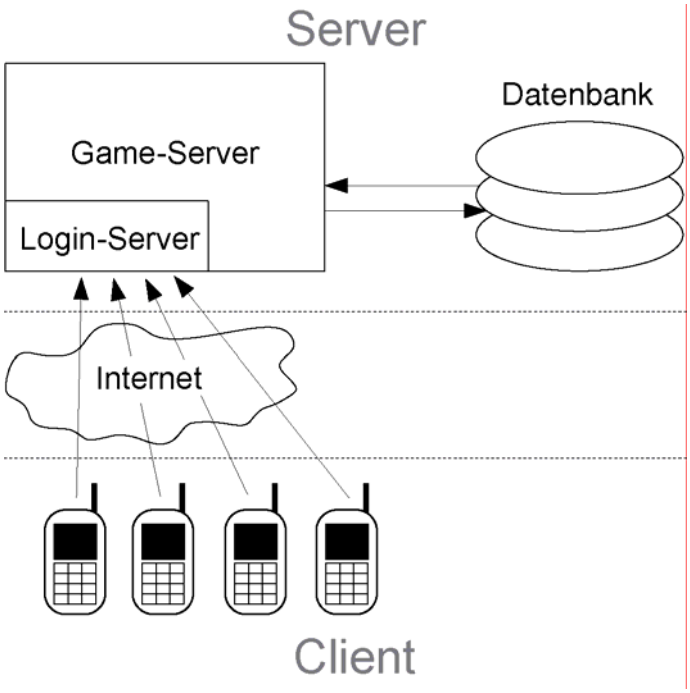

Abbildung 4-1: Minimale Client-Server-Architektur eines mobilen Rollenspieles

4.1.2 Einschränkungen zur Umsetzung in dieser Arbeit

Der Fokus dieser Arbeit liegt auf der clientseitigen Entwicklung eines Frameworks. Die serverseitigen Technologien werden hier nicht realisiert. Es steht nur zu Demonstrationszwecken des Clients ein kleiner Server zur Verfügung, der Testdaten zur Präsentation generieren und an den Client senden kann.

4.2 Anforderungskatalog für das Framework

In diesem Abschnitt wird der Anforderungskatalog für das Framework vorgestellt. Neben den funktionalen Anforderungen in Abschnitt 4.2.1 werden in Abschnitt 4.2.2 die technischen und in Abschnitt 4.2.3 die nicht-funktionalen Anforderungen an das Framework definiert.

4.2.1 Funktionale Anforderungen

In diesem Kapitel werden die sich aus der in Kapitel 3 durchgeführten Analyse ergebenden funktionalen Anforderungen an das Framework zusammengefasst.
Die funktionalen Anforderungen werden in zwei Gruppen geteilt. Abschnitt 4.2.1.1 umfasst zuerst die Kernfunktionen des Frameworks, in 4.2.1.2 werden dann die variablen Funktionen vorgestellt.

4.2.1.1 Kernfunktionen

Aus den in Kapitel 3.2 ermittelten Eigenschaften eines mobilen Rollenspiels ergeben sich folgende Kernfunktionen, die jedes mobile Rollenspiel haben muss und die damit die minimale Schnittmenge bilden:

- Zwei Alternativen für die Darstellung der Position des eigenen Charakters anhand der ermittelten Geodaten:
    - Auf der *Google Maps*™-Karte im realen Kontext (siehe 3.3)
    - Auf einer virtuellen Karte in einem virtuellen Kontext (siehe 3.3)
- Login und Anmeldung bei einem Game-Server (Einloggen mit Benutzername und Passwort, um dann den eigenen Charakter auszuwählen oder einen neuen zu generieren)
- Generieren eines neuen Charakters (Auswahl der je nach Spiel unterschiedlichen Eigenschaften, Rassen oder Klassen)

Zusätzlich beinhalten viele Online-Rollenspiele die Registrierung eines Benutzer-Accounts. Dies ist die Registrierung bei der Firma, welche das Spiel betreibt und enthält bei kostenpflichtigen Spielen die Auswahl der Zahlungsmethode. Diese Funktion wird sehr individuell, z.B. über Webservices auf den Webseiten der Betreiber, realisiert und ist daher nicht Inhalt dieser Arbeit.

4.2.1.2 Variable Funktionen

Folgende in Kapitel 3.2 ermittelten Eigenschaften müssen für den Entwickler eines mobilen Rollenspiels entweder optional sein, da sie nicht in jeder denkbaren Spielausprägung benötigt werden[23], oder werden je nach Spiel sehr unterschiedlich implementiert. Für diese Funktionen muss das Framework nur eine Schnittstelle anbieten und einige Basisfunktionen implementieren. Der Entwickler kann seine Anwendung dann aus einzelnen Bausteinen zusammensetzen, für die er jeweils die gewünschten Funktionen implementiert.
Diese Eigenschaften sind:

- Anzeige, Auswertung, Speicherung und Generierung von Charakterinformationen aller Art
- Betreten und Verlassen virtueller Gebäude
- Darstellen und Ausführen von Bewegungsroutinen für Monster
- Dialog mit Nicht-Spieler-Charakteren
- Handel zwischen Spielern
- Visualisieren und Überprüfen von Quests
- Kampf gegen Monster und Spieler

4.2.2 Technische Funktionen

Dieses Kapitel stellt die für die Entwicklung des Frameworks verwendeten Technologien und die daraus resultierenden technischen Voraussetzungen vor.

- **Plattform**
  Als Plattform wird *Google™ Android* (siehe Kapitel 2.5) verwendet. Entwickelt wird in der aktuellen SDK-Version 2.0.1.
  Alle jedoch hier bisher erörterten theoretischen Konzepte zu den Eigenschaften mobiler Rollenspiele können mit geringen Anpassungen auch auf andere Plattformen wie die des *Apple iPhone®* übertragen werden.
- **Netzwerkkommunikation**
  Die Kommunikation muss aufgrund des mobilen Ansatzes drahtlos sein. Da eine feste Verbindung zu einem Game-Server benötigt wird, finden Techniken zur direkten

[23] An Beispielen wie dem Online-Rollenspiel „A Tale in the Desert 4“ [eGenesis 2009], in dem es keine Kampfmöglichkeiten gibt, welches aber trotzdem eindeutige Rollenspiel-Eigenschaften aufweist, lässt sich gut erkennen, wie breit der Rahmen denkbarer Spielkonzepte ist.

Übermittlung von Daten über kurze Distanzen zwischen Mobiltelefonen wie *Bluetooth®* keine Verwendung. Stattdessen wird wie in den Online-Rollenspielen über eine TCP-bezogene Socket-Verbindung eine Verbindung zum Game-Server aufgebaut (siehe 2.6.2).

- **Positionsbestimmung durch *GPS***
  Für die Positionsbestimmung soll das von *Android* direkt unterstützte *GPS* (siehe Kapitel 2.4) verwendet werden.
- **Mobiltelefon**
  Das zur Ausführung der Software verwendete Mobiltelefon muss als Betriebssystem *Android* verwenden, eine z.B. über *UMTS*™ gestützte Internetverbindung aufbauen können und Zugriff auf einen (eingebauten oder externen) *GPS*-Empfänger haben.

4.2.3 Nicht-Funktionale Anforderungen

Neben den bereits unter 4.2.2 vorgestellten technischen Voraussetzungen gelten für dieses Framework außer den in der Softwareentwicklung typischen Forderungen nach Zuverlässigkeit, Ergonomie und Wartbarkeit folgende nicht funktionale Anforderungen:

- **Hintergrundfunktion**
  Das Spiel, speziell die ortssensitiven Funktionen und die Netzwerk-Kommunikation laufen im Hintergrund weiter, auch wenn der Spieler mit anderen Programmen arbeitet oder mit Telefonfunktionen beschäftigt ist.
- **Flexibilität**
  Soweit kein logischer Bezug besteht, sollen alle unter 4.2.1.2 genannten Funktionen vom Nutzer des Frameworks unabhängig voneinander verwendet werden können.
- **Dokumentation**
  Da das Framework als Grundlage für die Entwicklung mobiler Rollenspiele dienen soll, ist eine ausführliche Dokumentation aller Klassen notwendig.

4.3 Entwurf des Basissystems

In diesem Kapitel wird das Basissystem entworfen. Der Abschnitt 4.3.1 gibt eine allgemeine Einführung, während im Abschnitt 4.3.2 die dreiteilige Komponentenstruktur des Entwurfs

vorgestellt wird. Der Abschnitt 4.3.3 stellt weitere Elemente des Frameworks vor. Im abschließenden Abschnitt 4.3.4 wird der entworfene Systemstart des Frameworks vorgestellt.

4.3.1 Allgemein

Da das zu entwickelnde Framework eine spezielle Aufgabe eines Anwendungsbereiches realisieren soll, handelt es sich um ein vertikales Framework (siehe 2.3). Es wird als White-Box-Framework konzipiert, welches hauptsächlich aus abstrakten Klassen besteht. Diese müssen vom Nutzer des Frameworks durch das Bilden konkreter Unterklassen spezialisiert werden. Der Kontrollfluss wird vom Framework gesteuert.

4.3.2 Komponenten

In diesem Kapitel werden die einzelnen Komponenten des Entwurfes vorgestellt. Abschnitt 4.3.2.1 gibt einen Überblick über die dreigeteilte Komponentenstruktur, die dann in den Abschnitten zur Netzwerkschicht (4.3.2.2), Applikationsschicht (4.3.2.3) und zur Darstellungsschicht (4.3.2.4) im Detail vorgestellt werden.

4.3.2.1 Überblick

Um die unter 4.2.1.1 aufgeführten Kernfunktionen des Frameworks umzusetzen, wurde ein dreigeteilter Entwurf konzipiert (siehe Abbildung 4-2). Eine Darstellungsschicht realisiert die grafische Darstellung für den Nutzer und nimmt dessen Eingaben entgegen. Der Kern der Anwendung liegt in der Applikationsschicht, in der alle persistenten Spielinformationen zwischengespeichert werden und ggf. nötige Berechnungen durchgeführt werden. Die Netzwerkschicht realisiert schließlich für die Applikationsebene den Zugriff über das Internet auf den Spielserver.

Abbildung 4-2: Komponentendiagramm des Entwurfes

4.3.2.2 Netzwerkschicht

Die Netzwerkschicht des Frameworks besteht aus drei abstrakten Klassen. Diese werden im Folgenden vorgestellt.

AbstractNetworkConnectionService

Um den Anforderungen der Hintergrundfunktion der mit dem Framework entwickelten Spiele gerecht zu werden, muss das Versenden und Empfangen der Daten in einem vom restlichen Kontrollfluss der Anwendung unabhängigen Dienst realisiert werden. Außerdem muss diese Netzwerkfunktionalität auch dann weiterlaufen, wenn die Anwendung durch andere Anwendungen überdeckt wird (z.B. durch einen eingehenden Telefonanruf), ohne beendet worden zu sein. Dies geschieht in *Android* mit Hilfe eines Services.

Die Klasse `AbstractNetworkConnectionService` spezialisiert einen von der Klasse `android.app.Service` abgeleiteten Service.

Eingehende Nachrichten (Client ← Server)

In einem eigenen Thread werden die vom Server versendeten und auf einer Socket-Verbindung eingehenden Daten empfangen. Diese werden dann per Broadcast-Intent verschickt und von der Klasse `AbstractProtocol` empfangen.

Senden von Nachrichten (Client → Server)

Für die Kommunikation zum Server wird der aktuelle Anwendungskontext – also die gerade aktive Activity – an den Service gebunden. In diesem können dann von außerhalb des Services Methoden im Service ausgeführt werden. Diese Technik wird verwendet, um den Subklassen von AbstractProtocol die Möglichkeit zum Aufrufen von Methoden des Services zu geben und somit Nachrichten an den Server zu senden. Dies geschieht hier mit Hilfe eines Objekts der Klasse `NetworkConnectionServiceSingleton`, welche die Schnittstelle `android.content.ServiceConnection` implementiert. Um sicherzugehen, dass alle Klassen sich an das gleiche Service-Connection-Objekt binden, unabhängig von sich ändernden Kontexten, beruht diese Klasse auf dem Singleton-Muster[24].

[24] Mit diesem Entwurfsmuster kann gesichert werden, dass es von einer Klasse nur eine Instanz in einer Anwendung geben kann.

AbstractProtocol

Die Protokoll-Klasse kapselt die Kommunikation zum Server gegenüber der Applikationsebene und hat Kenntnis über die zugrunde liegende Struktur des Netzwerkverkehrs.

Als Sub-Klasse der Klasse `android.content.BroadcastReceiver` empfängt das Protokoll die vom Service (s. o.) versendeten Intents mit den eingehenden Netzwerknachrichten, analysiert diese und verarbeitet die enthaltenen Informationen selber oder delegiert diese weiter in die Applikationsebene.

Für die umgekehrte Kommunikation stellt das Protokoll Funktionen zur Verfügung, um Informationen aus der Applikationsebene an den Service zur Übertragung zum Server zu übergeben. Dafür werden die zu übertragenen Daten auf Basis der verwendeten Kommunikationsvereinbarung codiert und dann mit Hilfe der Klasse `NetworkConnectionServiceSingleton` über einen Methodenaufruf im Service an jenen weitergegeben.

AbstractConnectionBroadcastReceiver

Diese Klasse empfängt die vom *Android* Connectivity Manager (siehe 2.5.2) versendeten Nachrichten über den Verbindungszustand des Netzwerkes. Der `AbstractConnectionBroadcastReceiver` und dessen Sub-Klassen steuern somit die Reaktion der Anwendung auf eine nicht mehr bestehende Internetverbindung. Die aktuelle Instanz des `AbstractNetworkConnectionService` wird vom `AbstractConnectionBroadcastReceiver` je nach Art der Nachricht gestartet oder beendet.

Die Abbildung 4-3 zeigt am Beispiel eines einfachen Login-Vorganges die interne Kommunikation der Klassen in der Netzwerkschicht.

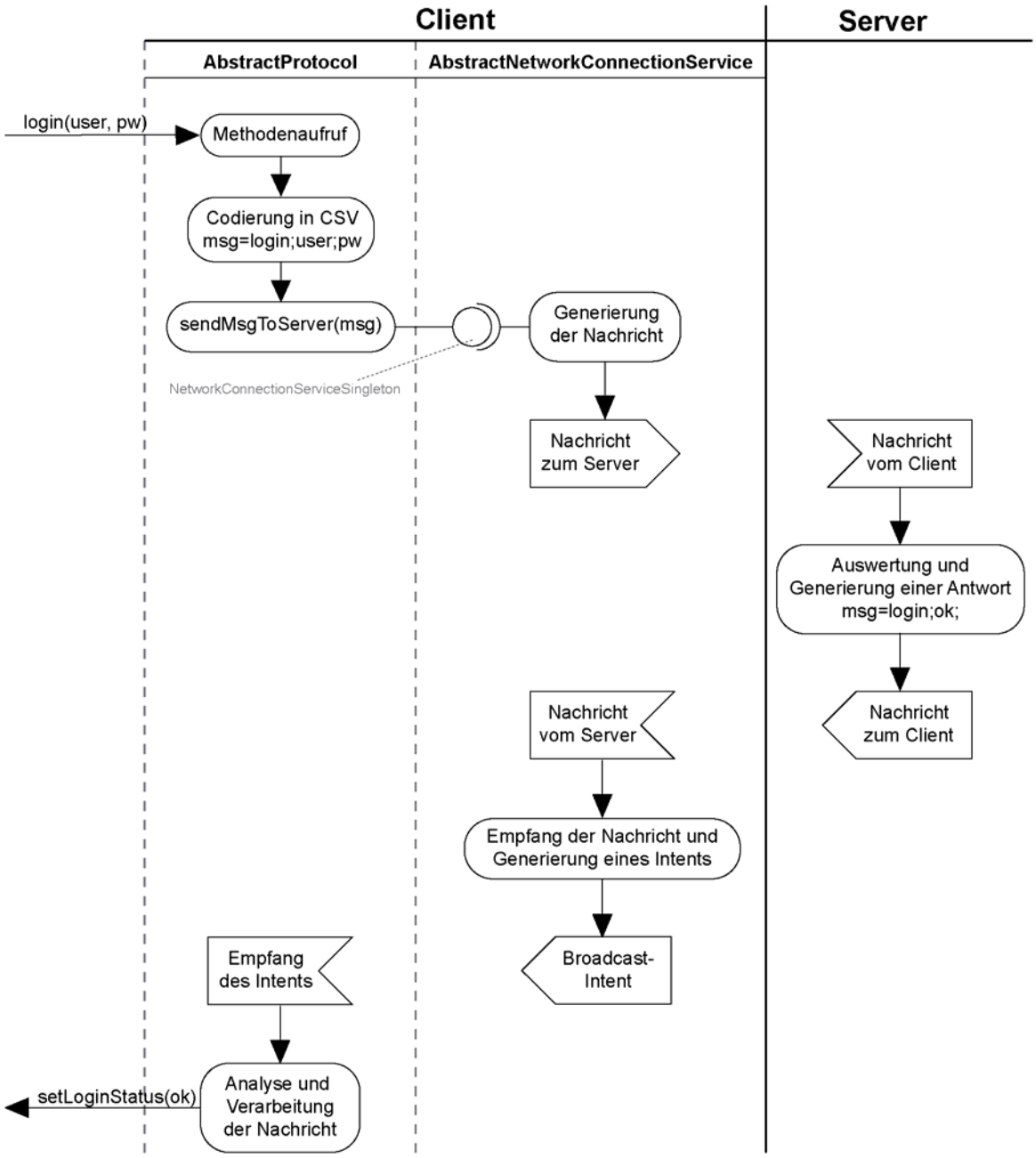

Abbildung 4-3: Darstellung der Kommunikation innerhalb der Netzwerkschicht am Beispiel eines Login-Vorganges (CSV-Format)

4.3.2.3 Applikationsschicht

Die Applikationsschicht wird von der Klasse `ApplicationManager` verwaltet. Da in *Android* sowohl das Binden an einen Service als auch die Registrierung bei einem Broadcast-Receiver abhängig von dem jeweiligen Kontext der Activity ist (repräsentiert durch die Klasse `android.content.Context`), ist eine Trennung der drei Komponenten Netzwerk, Applikation und Darstellung nicht vollständig möglich. Der Application-Manager muss daher neben den Klassen aus der eigenen Schicht auch Klassen der Netzwerk- und

Darstellungsschicht kennen, um die Durchführung oben genannter Funktionen zu ermöglichen. Er dient damit als zentraler Vermittler zwischen den Schichten.
Im Folgenden werden die beiden zentralen abstrakten Klassen der Applikationsschicht vorgestellt.

AbstractClientDataContainer
Diese Klasse ist ein Container für alle clientseitig zwischenzuspeichernden Daten. Dazu zählen neben dem Login-Status auch die Benutzernummer und der Name des Nutzers. In den abgeleiteten Klassen können vom Entwickler eines konkreten Spieles z.B. verwendete Charakterdaten, Nicht-Spieler-Charaktere u. Ä. gespeichert werden.

AbstractLocationDataManager
Der `AbstractLocationDataManager` ist für die Positionsbestimmung des Endgerätes zuständig. Dies geschieht über die *GPS*-Schnittstelle des *Android*-Systems. Der so bestimmte Geopunkt kann je nach Anwendungsfall dann in der Subklasse ausgewertet werden. Über den `ApplicationManager` besteht Zugriff auf das Protokoll und damit auf die Netzwerkschicht, um die ermittelte neue Position dem Server mitteilen zu können.

4.3.2.4 Darstellungsschicht

Jede darzustellende Bildschirmseite ist in *Android* eine Ableitung der Klasse `android.app.Activity`. Das Framework stellt daher eine `AbstractMobileRPGActivity` zur Verfügung, die als Superklasse der meisten Activities dienen soll. Eine Ausnahme stellen die Klassen `AbstractGoogleMapActivity` und `AbstractPreferenceActivity` dar. Jene bilden Subklassen von `com.google.android.maps.MapActivity` bzw. `android.preference.PreferenceActivity`. Beide genannten Klassen der *Android*-API sind jedoch auch Subklassen von `android.app.Activity`.

Alle abstrakten Klassen in der Darstellungsschicht haben gemeinsam, dass sie eine grundlegende Verarbeitung des *Android*-Live-Cycles (siehe Kapitel 2.5.2 zum Activity Manager) sicherstellen, sich beim Broadcast-Receiver für die Überprüfung der Internetverbindung registrieren und außerdem beim Starten die Methode generateApplicationManager() (wie Kapitel 4.3.4 näher erläutert) aus einer Spezialisierung

der `AbstractApplicationFactory` aufrufen. Es folgt eine Auflistung der Klassen und ihrer Einsatzgebiete. Die oben genannten Funktionen werden nicht erneut aufgeführt.

AbstractGoogleMapActivity

Diese Klasse dient als Superklasse für die Darstellung der Spielwelt mit der Karte von *Google Maps*™. Damit ist sie die zentrale Activity in mobilen Rollenspielen.

AbstractPreferenceActivity

Diese Klasse muss zur Implementierung einer Preference-Activity, die zur Steuerung von anwendungsspezifischen Einstellungen dient, verwendet werden.

AbstractMobileRPGActivity

Diese Klasse sollte Superklasse für alle Activities sein, die nicht in eine der vorher genannten Kategorien fallen.

4.3.3 Weitere Elemente des Frameworks

Dieses Kapitel beschreibt in Abschnitt 4.3.3.1 die Konzeption der Kommunikation zwischen Darstellungs- und Applikationsschicht.

4.3.3.1 Kommunikation zwischen Darstellungs- und Applikationsschicht

Es ist in *Android* nicht möglich, darstellende View-Elemente (siehe 2.5.2) einer Activity aus einem anderen Thread zu verändern, als dem, in dem die Activity läuft. Die in den Views darzustellenden Informationen sind jedoch in der Applikationsschicht gespeichert. Eine Möglichkeit dieses Problem zu lösen, ist die Verwendung eines Threads in der Darstellungsschicht, der in kurzen Abständen per Polling die Applikationsschicht nach neuen Daten abfragt. Diese, dann in dem Thread der Darstellungsschicht vorliegenden Daten, können mit Hilfe eines Objekts von `android.os.Message` zu einem `android.os.Handler` in den Thread verschickt werden in dem die Views laufen. Dort ist es dann möglich, mit den Daten die Views zu aktualisieren.

Das Framework stellt dafür zwei abstrakte Klassen zur Verfügung:

AbstractGuiUpdateTask

Diese Klasse ist Subklasse von `java.util.TimerTask` und zuständig für das Polling in der Applikationsebene und das Verpacken und Versenden der Daten in das Message-Objekt.

AbstractGuiUpdateMessageHandler
Diese Klasse ist für den Empfang der Nachrichten und das Aktualisieren der GUI zuständig.

4.3.4 Start des Systems

Die zum Framework gehörigen abstrakten Activities rufen in ihrer onCreate()-Methode die Methode generateApplicationManager() aus einer referenzierten Subklasse von `AbstractApplicationFactory` auf. In dieser Subklasse muss die gesamte Darstellungs- und Netzwerkschicht erzeugt werden. Da die Methode generateApplicationManager() beim Erzeugen jeder Activity aufgerufen wird, ist die Factory[25] als Singleton implementiert, so dass es zu jedem Zeitpunkt nur eine Instanz der Klasse `ApplicationManager` geben kann.

4.4 Erweiterter Entwurf für die funktionalen Anforderungen

Dieser Abschnitt stellt die individuellen Möglichkeiten zur Integration der funktionalen Anforderungen (siehe 4.2.1) und die sich gegebenenfalls daraus ergebenden Erweiterungen des Entwurfs vor.
Der Abschnitt 4.4.1 behandelt die Kernfunktionen, der Abschnitt 4.4.2 die variablen Funktionen der Anforderungsliste.

4.4.1 Kernfunktionen

Dieses Kapitel beschreibt die Konzeption der Kernfunktionen der funktionalen Anforderungen. Von Auswahl von virtuellem und realem Kontext (4.4.1.1) über die Umsetzung der Login-Funktion (4.4.1.2) wird auf die Charaktergenerierung (4.4.1.3) eingegangen.

[25] Die Fabrikmethode ist ein Entwurfsmuster, in dem die Erzeugung von Objekten an Subklassen delegiert wird.

4.4.1.1 Auswahl von realem und virtuellem Kontext

Eine Realisierung des realen Kontextes ist über Subklassen von `AbstractGoogleMapActivity` gewährleistet. Da für eine Umsetzung der virtuellen Karte im virtuellen Kontext die *Google Maps*™-Karte nicht benötigt wird, kann dies über eine abgeleitete Klasse von `AbstractMobileRPGActivity` realisiert werden. In dieser muss dann über Instanzen von `android.widget.ImageView` die virtuelle Karte angezeigt werden. Alle weiteren Schnittstellen und Funktionen sind nicht beeinflusst.

4.4.1.2 Server-Login

Auf Clientseite ist eine Login-Funktion mit Hilfe der Netzwerkverbindung und einer Speicherung des Login-Status in einer Instanz der Klasse `AbstractClientDataContainer` realisierbar. Hierfür ist keine weitere Unterstützung des Frameworks notwendig, da die komplette Intelligenz der Authentifizierung auf Serverseite liegt.

4.4.1.3 Charaktergenerierung

Der Entwickler eines mobilen Rollenspiels kann die individuelle Charaktergenerierung seines Spiels mit Hilfe des Frameworks realisieren. Die von Spiel zu Spiel unterschiedlichen Anforderungen an diese Funktion, bilden nur eine sehr kleine Schnittmenge, die im Kern aus der Netzwerkübertragung besteht. Auch hier liegt die zentrale Logik der Korrektheitsprüfung auf Serverseite, um Manipulationsversuchen vorzubeugen (siehe 2.6.3). Daher macht es hier keinen Sinn, eine eigene abstrakte Klasse als Basis für die Charaktergenerierung anzubieten. Mit Hilfe der Activity `AbstractMobileRPGActivity` kann eine individuelle Oberfläche erzeugt werden, die Zugriff auf das Netzwerk hat und so die Charaktererschaffung mit dem Server abstimmen kann.

4.4.2 Variable Funktionen

Dieses Kapitel beschreibt die Umsetzung der variablen Funktionen (siehe 4.2.1.2). Der erste Abschnitt 4.4.2.1 behandelt die Verwaltung von Charakterinformationen, im folgenden Abschnitt 4.4.2.2 werden die Bewegungsroutinen von Monstern behandelt. Der Abschnitt 4.4.2.3 handelt von der Kampfberechnung. Im vorletzten Abschnitt 4.4.2.4 wird die

Visualisierung und Überprüfung von Quests beschrieben. Der letzte Abschnitt 4.4.2.5 fasst drei Punkte zu einem Punkt Interaktion zusammen und beschreibt deren Konzeption.

4.4.2.1 Verwaltung von Charakterinformationen

Die Anzeige der Charakterinformationen kann über die unter 4.3.3.1 beschriebene Schnittstelle zwischen Applikations- und Darstellungsschicht realisiert werden. So können die in der Applikationsschicht gespeicherten Daten an die Darstellungsschicht übergeben werden und dort angezeigt werden. Die Anzeige ist individuell für jedes Spiel und daher ist eine weitere Konzeption des Frameworks an dieser Stelle nicht sinnvoll.
Die Speicherung der Daten erfolgt wie beschrieben in der Applikationsschicht in Subklassen der Klasse `AbstractClientDataContainer`. Die Generierung und Auswertung der Daten erfolgt serverseitig.

4.4.2.2 Darstellen und Ausführen von Bewegungsroutinen für Monster

Da die Berechnung der Bewegungen der Monster für alle Spieler identisch sein soll, erfolgt diese serverseitig und wird von dort an die Clients verteilt. Für die Darstellung dieser Monster müssen im realen Kontext Subklassen von `com.google.android.maps.Overlay` verwendet werden. Da das Aussehen dieser Grafiken auf der *Google Maps*™-Karte für jedes Spiel individuell ist, ist die Entwicklung eigener Overlay-Subklassen Aufgabe der Nutzer des Frameworks. Bei der Spezialisierung der Activity `AbstractGoogleMapActivity` muss die Methode updateOverlays() überschrieben werden, in der die Routine für die Aktualisierung aller Overlays implementiert werden muss.

4.4.2.3 Kampfberechnung gegen Monster und Spieler

Die Entscheidung, ob es zu einem Kampf kommt, muss serverseitig anhand der Positionsdaten der Charaktere, Monster und Nicht-Spieler-Charaktere gefällt werden. Aufgrund der mangelnden Präzision der *GPS*-Technologie (siehe 2.4.2) muss der Server die Bestimmung, ob Figuren benachbart sind, großzügig anlegen. Je nach zu erwartender Genauigkeit, können so z.B. schon Figuren im Abstand von 50 Metern als benachbart angesehen werden. Alle für den Kampf relevanten Berechnungen finden dann ebenfalls serverseitig statt, um die Chance auf Manipulation zu verringern. Der Client muss jedoch eine Oberfläche bereitstellen, auf der der eigentliche Kampf stattfinden kann. Hier sollte der Spieler dann seine speziellen Kampfaktionen und Zaubersprüche auswählen können.

Dies ist ebenfalls von Spiel zu Spiel unterschiedlich sowohl in der optischen Umsetzung der Benutzerschnittstelle als auch im genauen Ablauf des Kampfes aus spielmechanischer Sicht. Hinzu kommt, dass wie in Abschnitt 3.2.1.5 dargelegt, für dieses Framework eine sehr offene und flexible Umsetzung des Kampfes aufgrund der angesprochenen Problematik bezüglich der Bewegung eines Spielers während des Kampfes essenziell ist. Aufgrund dessen ist es die Aufgabe der Spieleentwickler, das bestehende Framework für ihre Bedürfnisse individuell zu erweitern.

4.4.2.4 Visualisieren und Überprüfen von Quests

Dieses Thema ist sehr vielschichtig. Es wird nicht nur von Rollenspiel zu Rollenspiel sehr unterschiedlich gehandhabt, sondern ist auch sehr eng verzahnt mit Elementen wie Dialogen mit Nicht-Spieler-Charakteren und Kämpfen. Aus diesem Grund bietet dieser Themenkomplex keine ausreichende Schnittmenge, die eine vorgefertigte Konzeption ermöglicht. Der Entwickler eines Spieles ist angehalten, hier durch Komposition einer Questkomponente an Subklassen von `AbstractClientDataContainer`, dem Spiel die Questfunktion hinzuzufügen. Alle Prüfungen und Berechnungen müssen dabei vom Server durchgeführt werden.

4.4.2.5 Interaktionen

Dieser Abschnitt fasst die Konzeption für die folgenden drei funktionalen Anforderungen zusammen:

- Betreten und Verlassen virtueller Gebäude
- Dialog mit Nicht-Spieler-Charakteren
- Handel zwischen Spielern

All diese Punkte benötigen einen gesonderten Dialog, müssen innerhalb dessen aber individuell umgesetzt werden. Eine gesonderte Konzeption einer abstrakten Klasse ist nicht notwendig, da ein individueller Dialog aus der Activity gestartet werden kann, die die *Google Maps*™-Karte implementiert. Diese neue Klasse wird dann über eine eigene Subklasse des `AbstractGuiUpdateMessageHandler` aktualisiert. Die Daten dafür werden an den Handler von einer für die Dialogseite individuellen Implementierung des `AbstractGuiUpdateTask` versendet.

4.5 Entscheidungen

In Unterabschnitt 4.5.1 wird eine Konzeptionsentscheidung bezüglich der Back- und Home-Tasten auf *Android*-Handys beschrieben.

4.5.1 Back- und Home-Taste

Android-Handys verfügen über eine Home- und eine Back-Taste. Das Framework soll sich so eng wie möglich an das von *Google* ™ für *Android*-Applikationen angestrebte Look & Feel[26] halten. Das Framework verändert daher nicht die Wirkung der Home-Taste, die beim Drücken die aktuelle Activity pausiert und zum Hauptbildschirm des Betriebssystems wechselt. Durch ein erneutes Starten des mobilen Rollenspiels wird die alte Activity reaktiviert und das Spiel an der ursprünglichen Stelle fortgesetzt.

Hingegen überschreibt das Framework in der `AbstractGoogleMapActivity`, die Methode onKeyDown() für die Back-Taste. Durch das leere Überschreiben der Funktion ignoriert das Framework alle Tastendrücke auf der Back-Taste. Dadurch wird die Eigenschaft des *Android*-Livecycles aufgehoben, die dafür sorgt, dass *Android* die durch Drücken der Back-Taste verlassenen Activities beendet und nicht nur pausiert. Dies ist für das mobile Rollenspiel unerwünscht, da das Spiel auch dann weiterlaufen soll, wenn der Spieler es nur in den Hintergrund bewegt. Das Framework bietet für echtes Beenden des Spiels eine Logout-Funktion im Menü an, mit der die Verbindung getrennt wird und der Spieler zurück auf den Startbildschirm kommt. Wird das Spiel von dort mittels der Home-Taste in den Hintergrund bewegt und nicht wieder gestartet, wird die laufende Applikation bei Speicherbedarf vom Activity Manager (siehe 2.5.2) beendet und die verwendeten Ressourcen freigegeben.

[26] Informationen zu der von *Google*™ vorgesehenen Funktionen der Home- und Back-Taste befinden sich unter: http://developer.android.com/intl/de/guide/practices/ui_guidelines/activity_task_design.html

5. Realisierung des Frameworks

In diesem Kapitel wird die für die Realisierung des Frameworks verwendete Hardware (Abschnitt 5.1) und Software (Abschnitt 5.2) vorgestellt.

5.1 Hardware

Für die Entwicklung des Frameworks stand kein Handy mit *Android* zur Verfügung. Daher wurde die Software vollständig auf dem von *Google*™ zur Verfügung gestellten Emulator getestet. Verwendet wurde die aktuelle Version 2.0.1. Als Hostsystem wurde ein *Intel® Core™ 2 Duo* P8600 CPU mit 2,4 GHz und 3 GB RAM verwendet.

5.2 Software

Dieser Abschnitt stellt die verwendete Software vor. Zuerst in Abschnitt 5.2.1 die verwendeten Versionen von *Android*, im Folgenden 5.2.2 die verwendete Entwicklungsumgebung und im abschließenden Abschnitt 5.2.3 eine weitere verwendete Bibliothek vor.

5.2.1 *Android*

Während des Entwicklungsprozesses wurde immer die aktuellste Version des *Android* SDK verwendet. Zum finalen Zeitpunkt dieser Arbeit ist das Version 2.0.1 release 1. Diese Version verwendet API Level 6. Das entwickelte Framework ist jedoch abwärtskompatibel bis API Level 3, also *Android* Version 1.5.
Für das *Android* DDMS und die *Android* Development Tools wurde ebenfalls die aktuellste Version 0.9.5 verwendet.

5.2.2 Entwicklungsumgebung

Für die Entwicklung des Frameworks wurde die Entwicklungsumgebung *Eclipse* in der Version *Galileo 3.5.0* genutzt. Die zugrunde liegende *Java*™ IDE war *Java EE*™ 1.2.0.

Eclipse bietet aufgrund des Plugins *Android* Development Tools (ADT) eine direkte Unterstützung bei der Entwicklung in *Android*. Neben der direkten Nutzung des Emulators aus *Eclipse*, wird auch eine direkte Steuerung des DDMS ermöglicht.

5.2.3 Weitere Bibliotheken

Zusätzlich zum regulären *Android* SDK wurde das Paket `com.google.android.maps` von *Google*™ verwendet. Dieses beinhaltet alle Klassen zur Darstellung der *Google Maps*™-Karte inklusive Funktionen zum Scrollen und Zoomen. Außerdem stellt das Paket die benötigten Klassen zur Auswertung der Geopunkte und der Anzeige von Overlays auf der Karte zur Verfügung.

6. Realisierung eines Spiels mit dem Framework

In diesem Kapitel wird die Realisierung eines Spiels mit dem in Kapitel 4 konzipierten Framework beschrieben.
Als erstes wird in Abschnitt 6.1 die verwendete Hard- und Software und deren Einschränkungen vorgestellt. Der Abschnitt 6.2 geht auf die realisierten Funktionen des Prototyps und die dabei gefällten Entwurfsentscheidungen ein. Das dann folgende Kapitel 6.3 beschreibt die für das Testszenario verwendete Servertechnologie und deren Protokoll. Im vorletzten Abschnitt 6.4 werden die Benutzerschnittstellen und Funktionen des Prototyps an Screenshots vorgestellt und beschrieben. Das letzte Kapitel 6.5 stellt eine Realisierungsmöglichkeit für den virtuellen Kontext vor.

6.1 Hardware und Software

Die für die Realisierung des Spiels verwendete Hard- und Software ist mit der für das Framework verwendeten identisch (siehe 5.1 und 5.2). Dieses Kapitel ergänzt die Informationen aus dem Kapitel um einige weitere Eigenschaften.

6.1.1 Netzwerkverbindung im Emulator

Der Client und der Server wurden für diese Arbeit zeitgleich am selben Rechner betrieben. Der von *Google*™ bereitgestellte Emulator läuft auf den Hostsystemen hinter einem virtuellen Router, ohne auf andere Emulatoren oder den Hostrechner Zugriff zu haben. Der Zugriff erfolgt stattdessen über ein lokales Netzwerk innerhalb des Hostrechners. Auf die Entwicklung im Rahmen dieser Arbeit hat das keinen Einfluss. Der Aufbau einer TCP-Verbindung ist über lokale IP-Adressen möglich.

6.1.2 Einschränkungen

Dieser Abschnitt geht auf die während der Realisierung auftretenden Einschränkungen bei der Simulation von *GPS*-Daten (6.1.2.1) und beim Verwenden mehrerer Emulatoren (6.1.2.2) ein.

6.1.2.1 *GPS*-Simulation

Die Versorgung des Emulators mit fiktiven *GPS*-Daten wurde über das *Eclipse*-Plugin DDMS gewährleistet. Neben der manuellen Übergabe einzelner Datensätze können ganze Routen im GPX- oder KML-Format vom Emulator verarbeitet werden. Für diese Arbeit wurde ein Rundgang durch Hamburg-Allermöhe mit 2480 Geopunkten auf 7,18 km im GPX-Format[27] verwendet.

Dies hat zwar zur Folge, dass sich die Spielfigur des Spielers wirklichkeitsnah wie in einem realen Fußmarsch bewegt, was eine gute Testumgebung für ein mobiles Rollenspiel liefert, für den Entwickler ist die Einflussnahme auf die Bewegungsabläufe in diesem Fall jedoch gering.

6.1.2.2 Parallele Emulatoren

Trotz der technischen Möglichkeit, an einem Rechner zwei oder mehr Instanzen eines *Android*-Emulators zu starten, gestaltete sich die Realisierung und das Testen einzelner Funktionen als schwierig. Um Kontakt zwischen Spielern testen zu können, müssten Georouten verwendet werden, welche sich kreuzen. Dafür wäre zusätzlich eine korrekte Synchronisation nötig. Aufgrund der zeitlichen Grenzen dieser Arbeit und des mit dieser Funktion verbundenen Aufwandes, wird auf Entwicklungen und Tests, die mehrere Emulatoren mit *GPS*-Daten verwenden, im Rahmen der Prototyp-Entwicklung verzichtet.

6.2 Prototyp

Dieses Kapitel stellt den im Rahmen dieser Arbeit entwickelten Prototypen vor. Neben einer allgemeinen Einführung (6.2.1) werden spezifische Entwurfsentscheidungen (6.2.2) und realisierte Funktionen (6.2.3) vorgestellt. Den Abschluss bildet der Abschnitt 6.2.4, der einige für diese Arbeit getroffene Vereinfachungen aufführt.

[27] Die GPX-Datei wurde entnommen von: http://www.gpsies.com/map.do?fileId=nupsipnumhsnqafn

6.2.1 Allgemein

Der Hauptteil der Prototyp-Realisierung sind die im bisherigen Verlauf der Arbeit herausgearbeiteten Kernfunktionen. Der hier entwickelte Prototyp soll aufzeigen, dass das in Kapitel 4 konzipierte Framework funktioniert und für den Anwendungsfall verwendet werden kann. Darüber hinaus soll exemplarisch gezeigt werden, wie das Framework um individuelle spielspezifische Funktionen unter Verwendung der durch das Framework bereitgestellten Möglichkeiten ergänzt werden kann. Der Prototyp erhebt nicht den Anspruch, ein spielbares mobiles Rollenspiel zu sein.

6.2.2 Entwurfsentscheidungen

In diesem Abschnitt werden Entwurfsentscheidungen zur Realisierung des Protoyps vorgestellt. Abschnitt 6.2.2.1 begründet die verwendete Menüsteuerung, Abschnitt 6.2.2.2 stellt das gewählte Szenario vor.

6.2.2.1 Menüsteuerung

Da Schaltflächen zur Spielsteuerung sehr viel Raum auf der Spielfläche in Anspruch nehmen, verwendet der Prototyp die von *Google*™ angebotenen Optionsmenüs zur Auswahl vieler Spielfunktionen. Dies ist kongruent mit der von *Google*™ im Look & Feel angedachten Anwendungssteuerung[28], hat jedoch den Nachteil, dass der Nutzer nicht immer alle Funktionen sofort sehen kann, da sich diese im Optionsmenü verbergen. Eine weitere Möglichkeit wäre die Nutzung von Kontext-Menüs, die andere Entwickler für ihre mobilen Rollenspiele verwenden könnten. Kontext-Menüs haben jedoch das Problem, dass sie für den Nutzer auf den ersten Blick nicht ersichtlich sind.

6.2.2.2 Szenario

Als Szenario wurde eine für Rollenspiele klassische Fantasywelt gewählt. Als Gegenstände treten mittelalterliche Waffen, Rüstungen und Helme auf. Die Monster sind u. a. Orks und Goblins.

[28] Hinweise zum Design von Menüs unter *Android* finden sich unter:
http://developer.android.com/intl/de/guide/practices/ui_guidelines/ menu_design.html

6.2.3 Funktionen

Exemplarisch wurden für den Prototyp einige Funktionen realisiert, die über die vom Framework gewährleisteten hinausgehen. Es folgt eine Aufzählung dieser Funktionen:

- **Login**
  Der Spieler kann sich über eine Login-Funktion am Server authentisieren.
- **Realisierungen im Optionsmenü**
  - Logout
  - Auswählen eines Einstellungsmenüs
  - Betreten von Gebäuden aus der Kartenansicht
  - Verlassen von Gebäuden
- **Darstellung charakterspezifischer Werte**
  Exemplarisch wurden einige Charakterwerte definiert und im Rahmen der Kartendarstellung angezeigt.
- **Bewegung von Monstern**
- Es werden vom Server übermittelte Monster-Bewegungen dargestellt.
- **Darstellung von Händlern**
- Die Position der Gebäude von Händlern wird auf der Karte angezeigt.
- **Erwerb von Gegenständen bei Händlern**
- In den Gebäuden der Händler gibt es die Möglichkeit verschiedene Gegenstände zu erwerben.

6.2.4 Vereinfachungen

Einige nicht zum Kernthema dieser Arbeit gehörende Programmieraufgaben wurden aufgrund des begrenzten Zeitrahmens vereinfacht.

- **Fehlerbehandlung**
  Mit Ausnahme der Socket-Verbindung fängt kein Programmteil Fehler ab. Ungültige Eingaben, falsche oder unvollständige Nachrichten vom Server, nicht gefundene Identifikationsnummern von Gegenständen, Monstern o.Ä. können zu Programmabstürzen oder falschen Zuständen führen.
- **Bildschirmdarstellung**

Neben der Tatsache, dass es keine einheitliche Bildschirmauflösung für Mobiltelefone mit *Android*-Betriebssystem gibt, ist diese Darstellung an *Android*-Handys zwischen Hoch- und Querformat wechselbar. Auf diese beiden unterschiedlichen Eigenschaften wurde bei der Entwicklung keine Rücksicht genommen.

6.3 Serveranbindung

Dieses Kapitel stellt zuerst in Abschnitt 6.3.1 die Eigenschaften des zum Testen des Prototyps verwendeten Servers vor, um in Abschnitt 6.3.2 auf das verwendete Protokoll für die Client-Server-Kommunikation einzugehen.

6.3.1 Server

Der zum Testen und Präsentieren der Software entwickelte Server ist eine einfache Java-Anwendung ohne Datenbankzugriff. Es wird also nur eine Persistenz der Daten gewährt, solange der Server angeschaltet ist. Alle generierten Spieldaten sind in Anzahl und Ausprägung an die spezifischen Bedürfnisse dieses Testszenarios angepasst. Geodaten von Gebäuden werden z.B. nur in dem Bereich in und um die verwendete GPX-Datei in Hamburg-Allermöhe generiert. Keine der erzeugten Spieldaten ist auf Spielbalance ausgelegt.

6.3.2 Protokoll

Das für die Kommunikation zwischen Client und Server verwendete Protokoll basiert auf dem CSV-Format. Alle Informationen werden in einer Zeichenkette hintereinander übertragen, nur durch Semikola getrennt. Eine Verschlüsselung des Datenstromes erfolgt nicht. Eine vollständige Auflistung des Protokolls ist im Anhang unter II zu finden.

6.4 Benutzerschnittstellen

Dieses Kapitel stellt die für den Prototyp entwickelten Benutzerschnittstellen vor. In allen Benutzerschnittstellen wurden die Standard-Views von *Android* verwendet. Auf grafisches Design wurde aufgrund dieser Prototyp-Realisierung keine Rücksicht genommen.

6.4.1 Startmenü

Abbildung 6-1 zeigt den Startbildschirm der Anwendung. Im unteren Bereich kann der Nutzer den Login-Bereich auswählen. An dieser Stelle könnte sich auch die Schaltfläche zur Generierung eines neuen Charakters befinden. Diese Funktion ist für diesen Prototyp jedoch nicht umgesetzt worden, da sie unabhängig vom Framework ist.

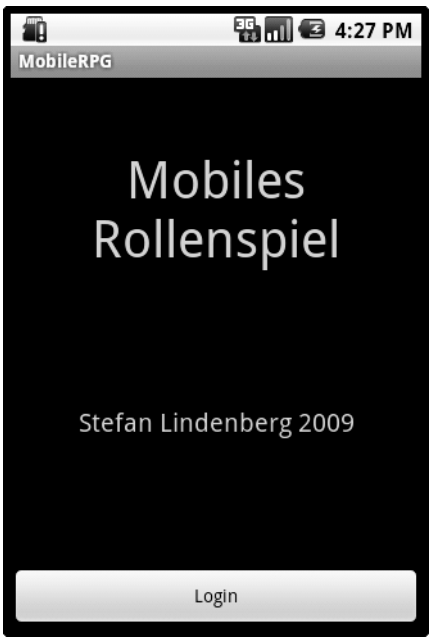

Abbildung 6-1: Startbildschirm der Anwendung

6.4.2 Loginmenü

Der in Abbildung 6-2 links dargestellte Login-Bildschirm zeigt die Eingabemaske für Benutzername und Passwort. Diese Abfrage wurde in einer eigenen Activity realisiert, die mit transparentem Rahmen oberhalb des Startbildschirmes liegt. Ein Drücken auf Login leitet den Authentisierungsvorgang ein, cancel führt auf den Startbildschirm zurück.

In der rechten Abbildung ist der während des Login-Vorganges erscheinende Dialog zu sehen. Der Client wartet auf eine positive Bestätigung des Servers, dass der eingegebene Benutzername und das eingegebene Passwort korrekt sind. Ist dies nicht der Fall, wird der Startbildschirm erneut dargestellt.

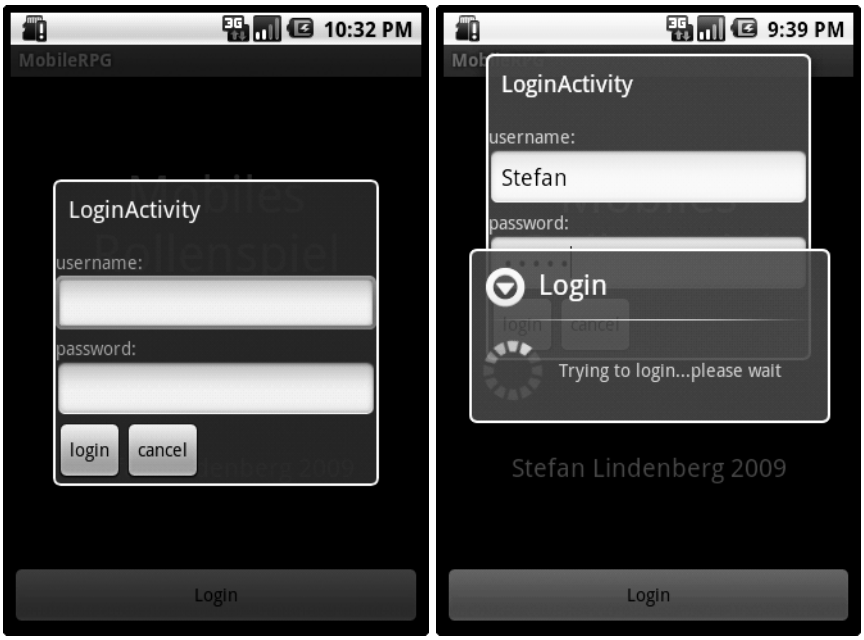

Abbildung 6-2: Login: Links Eingabemaske, rechts: Dialog während Client auf Antwort wartet

6.4.3 *Google*™-Karte

Der Kern des Spiels ist die Darstellung der *Google Maps*™-Karte. Abbildung 6-3[29] zeigt im oberen Bereich den Namen und vom Server übertragene Daten des Charakters. Einige Werte wie die Lebenspunkte unter dem Namen verändern sich. Hier simuliert der Server eine Regeneration der Lebensenergie des Spielers. Der Server berechnet die neuen Werte und überträgt diese an den Client, um sie anschließend auf diesem Bildschirm darzustellen.
Der Hauptbildschirm im unteren Bereich zeigt die Karte, zentriert um die eigene Spielfigur, dargestellt durch das Gesicht mit Helm. Unter der Figur ist eine kleine Hütte dargestellt, die das Gebäude eines Händlers repräsentiert. In der unteren linken Bildschirmecke ist das Symbol für ein Monster zu erkennen.

Folgende Symbole werden auf der Karte verwendet:

 eigener Charakter

 Haus eines Händlers

 Monster

[29] Alle im Prototypen benutzten Icons, Grafiken und die Kartenelemente der virtuellen Karte wurden mit freundlicher Genehmigung von Thorsten Lindenberg verwendet.

Abbildung 6-3: Hauptbildschirm mit Charakterdaten

6.4.3.1 Optionsmenü

Das in Abbildung 6-4 dargestellte Optionsmenü bietet drei Auswahlmöglichkeiten. Mit der Auswahl „Einstellungen“ wird eine spezielle Activity zur Konfiguration der Spieleinstellungen geöffnet (siehe 6.4.3.3).

Die Option „Gebäude betreten“ kann nur ausgewählt werden, wenn sich ein Spieler in direkter Umgebung zu einem Gebäude befindet. Ist das nicht der Fall ist der Auswahlpunkt sichtbar, aber deaktiviert. Ein Auswählen des Punktes öffnet eine Activity, welche die Optionen anzeigt, die dem Spieler in dem Gebäude zur Verfügung stehen. Um das Gebäude zu verlassen, muss erneut das Optionsmenü verwendet werden. Anstatt der Option „Gebäude betreten“ gibt es jetzt die Option „Gebäude verlassen“.

Die dritte Option hingegen sorgt für eine Abmeldung beim Server und eine Überleitung in den Startbildschirm. Wenn gewünscht kann sich der Spieler danach unter gleichem oder anderem Benutzernamen erneut anmelden.

Abbildung 6-4: offenes Optionsmenü

6.4.3.2 Kauf bei Händlern

Wenn ein Spieler das Gebäude eines Händlers betreten hat, wird die Auswahl der von dem Händler angebotenen Waren angezeigt. Dies ist in Abbildung 6-5 ersichtlich. Links befinden sich Grafiken der Waren, in der Mitte die Bezeichnungen und Spielfunktionen, rechts die Option, die Gegenstände zu kaufen, sofern der Charakter genügend Geld dabei hat. Neben einer Anzeige in der Charakter-Status-Leiste der Kartendarstellung, wird auch unter den vom Händler angebotenen Waren das zur Verfügung stehende Geld angezeigt. Alle Waren für die ausreichend Geld vorhanden ist können erworben werden, für alle anderen ist die Schaltfläche am rechten Bildschirmrand deaktiviert.

Entscheidet sich der Spieler für einen Gegenstand, wird diese Information zum Server übertragen. Dieser prüft erneut, ob der Spieler genügend Geld zur Verfügung hat. Sollte das der Fall sein, werden die neuen Charakterdaten mit dem reduzierten Geld und das neue um den gekauften Gegenstand verminderte Angebot des Händlers übermittelt. Ist der Kauf aus irgendeinem Grund doch nicht möglich, weil z.B. ein anderer Spieler schneller war und den Gegenstand zuerst erworben hat, kommt es zum Verfall der Kaufanfrage und es wird nur das neue Angebot übermittelt.

Die Implementierung eines Handelssystems in der hier getätigten Form verlangt noch nach vielen weiteren Funktionen, damit aus dem Prototyp ein wirkliches Spiel wird. Hier folgt eine Aufzählung ohne Anspruch auf Vollständigkeit:

- Anzeige der Gegenstände im Besitz des Charakters
- Unterscheidung zwischen Gegenständen, die getragen werden, und Gegenständen im Rucksack
- Prüfung auf Tragbarkeit der Gegenstände, z.B. nicht drei Waffen zeitgleich oder eine zweihändige Waffe und ein Schild
- Einfluss der Gegenstände auf Spielfaktoren wie z.B. Kampfwerte
- Auswahl der gekauften Menge bei Verbrauchsgütern wie Pfeilen
- Verkauf von Gegenständen
- Dynamische Preise nach Angebot und Nachfrage

Abbildung 6-5: Innenansicht eines Handels-Gebäudes

6.4.3.3 Einstellungsmenü

Im Einstellungsmenü (siehe Abbildung 6-6) sollten alle Einstellungsmöglichkeiten untergebracht werden, die der Spieler zur Verfügung hat. Exemplarisch kann hier das Satellitenbild an- bzw. ausgeschaltet werden. Die in dieser Arbeit gezeigten Screenshots zeigen die *Google Maps*™-Karte mit angezeigtem Satellitenbild.

Abbildung 6-6: Einstellungsmenü

6.5 Realisierung des virtuellen Kontextes

Dieses Kapitel skizziert die Realisierung des in Kapitel 3.3.2 beschriebenen virtuellen Kontextes. Der Abschnitt 6.5.1 beschreibt die grundlegenden Berechnungen, Abschnitt 6.5.2 die Realisierung der Darstellung und der letzte Abschnitt 6.5.3 schließlich die notwendige Kollisionskontrolle.

6.5.1 Berechnung

Nach dem Übermitteln einer *GPS*-Koordinate muss der Server die Differenz der Bewegung zum vorherigen Standpunkt in eine Längeneinheit umrechnen. Dieser Richtungsvektor wird dann auf die virtuelle Karte projiziert und so die neue Position berechnet. Diese wird in einem 2-dimensionalen Koordinatensystem (die vom Längen- und Breitengrad der Erde verwendete Darstellung in einem Gradnetz ist hier nicht mehr notwendig) auf die virtuelle Karte übertragen. Da für diese Zuweisung nur der alte (virtuelle) Standpunkt und der Richtungsvektor der letzten Bewegung relevant ist und nicht die aktuelle Position, können sich wie in Kapitel 3.3.2 beschrieben, mehrere Charaktere an einem virtuellen Ort befinden, auch wenn sich die Spieler weit voneinander entfernt befinden.

6.5.2 Darstellung

Für die Realisierung des virtuellen Kontextes kann die `AbstractGoogleMapActivity` nicht verwendet werden, da diese ausschließlich für die Darstellung der *Google Maps*™-Karte vorgesehen ist. Stattdessen muss in einer von `AbstractMobileRPGActivity` abgeleiteten Klasse ein schachbrettartiges 3x3 Raster von Instanzen von `android.widget.ImageView` erzeugt werden. In diesen kann die in etliche Einzelteile (z.B. à 500*500 px) unterteilte virtuelle Karte angezeigt werden. Wie in Abbildung 6-7 gezeigt, werden dort immer das Teilstück der Karte angezeigt auf dem sich der Charakter gerade befindet und alle acht angrenzenden Kartenteile. In dem Beispiel befindet sich der Charakter auf Kartenelement 3-3 und bewegt sich nach rechts. Wie auch im realen Kontext wird die Darstellung immer um den Standpunkt des Charakters zentriert. Im Laufe der Beispielbewegung verschiebt sich die Karte immer mehr nach rechts, es wird somit immer mehr von dem Zielteil 4-3 sichtbar. Sobald der Charakter die Grenze von Teil 3-3 und 4-3 überquert hat, werden alle sich in den Instanzen von `ImageView` dargestellten Grafiken um einen Schritt nach links geschoben. Die Teile 2-2, 2-3 und 2-4 rutschen damit links aus der Darstellung und werden durch die sich vorher in der Mitte befindliche Reihe ersetzt. Anstatt der Teile 4-2, 4-3 und 4-4 werden in der rechten Spalte nun 5-2, 5-3 und 5-4 angezeigt.

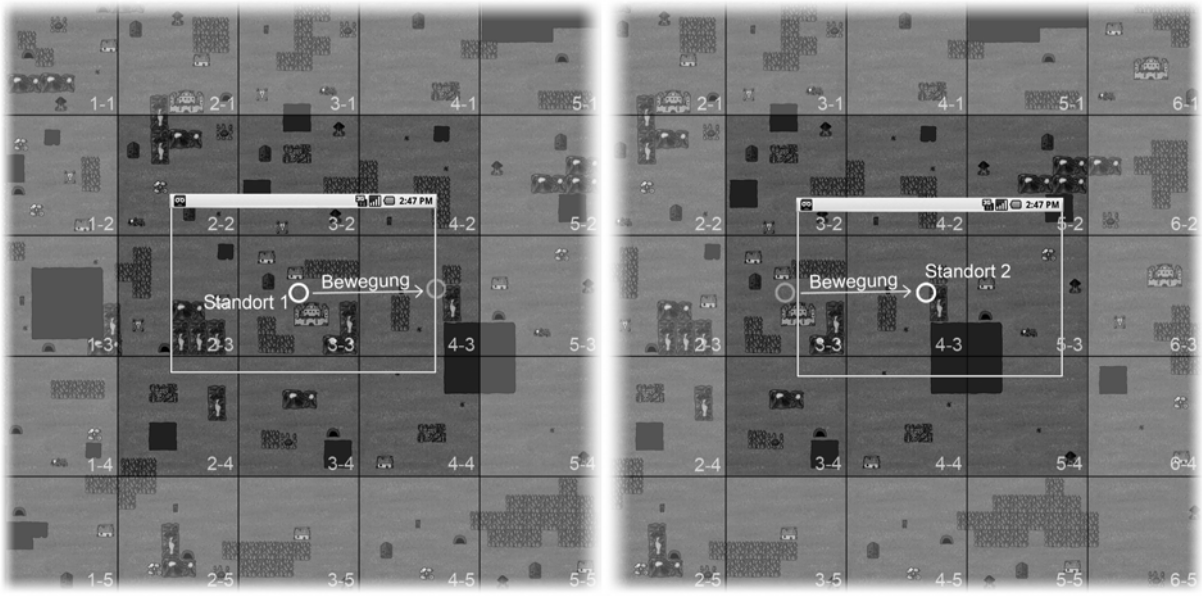

Abbildung 6-7: Bewegung auf einer virtuellen Karte

Dieses System hat zur Folge, dass unabhängig von der Drehrichtung des Bildschirms immer eine randlose Karte auf dem Bildschirm angezeigt wird und die Übergänge zwischen den

Kartenteilen fließend sind. Außerdem müssen jeweils nur neun Grafiken zeitgleich vom System im Speicher gehalten werden.

6.5.3 Kollisionskontrolle

Der Server muss nach Übertragung der neuen Bewegungsinformation des Clients nun eine Überprüfung auf deren Korrektheit machen. Sollte z.B. eine Bewegung den Kartenrand oder ein geographisches Hindernis wie ein Gewässer erreichen, sendet der Server als Zielkoordinate die Quellkoordinate des Charakters zurück. Eine Bewegung hat dann nicht stattgefunden.

7. Zusammenfassung und Fazit

Ziel dieser Arbeit war die Entwicklung eines Rollenspiels für *Android* unter Einbeziehung der Konzepte des Pervasive Gaming. Im Rollenspiel als Gesellschaftsspiel verkörpern die Spieler Charaktere mit individuellen Eigenschaften, um gemeinsam Abenteuer zu bestehen. Computer-Rollenspiele stellen die Umsetzung der Rollenspiele für Einzelspieler am Computer dar, während in Online-Rollenspielen mehrere Spieler zeitgleich zusammen spielen können. Pervasive Games zeichnen sich durch Ortssensitivität aus, so dass der Spieler durch seine eigene reale Position die Position der Spielfigur bestimmt. Aus den Eigenschaften von Rollenspielen und denen der Pervasive Games wurde ein neues Spielkonzept der mobilen Rollenspiele herausgearbeitet.
Basierend auf dieser Grundlage sollte nun ein clientseitiges Framework entwickelt werden, welches die Schnittmenge denkbarer Ausprägungen der neuen Spielform abdeckt. Dabei zeichnete sich früh ab, dass es aufgrund unterschiedlichster Spielkonzepte nur für wenige Funktionen Überschneidungen gibt. Außerdem muss zur Gewährleistung der Persistenz und zur Verhinderung von Manipulation der Großteil der Spiellogik auf Seiten des Servers liegen. Der Client übernimmt daher nur die Darstellung der Spieldaten und die Interaktion mit dem Nutzer. Somit muss das Framework die Möglichkeiten für das Hinzufügen verschiedenster Spielfunktionen in spielindividuellen Ausprägungen bieten. Den Kern bildet aber der Aufbau und die Aufrechterhaltung der Netzwerkverbindung und das Entgegennehmen der *GPS*-Daten. Entsprechend gestaltete sich die Konzeption des Frameworks: Dieses gewährleistet die clientseitige Serveranbindung, den Zugriff auf die *GPS*-Schnittstelle, die Datenverwaltung der übertragenen Weltdaten und steuert den Kontrollfluss über eine zentrale Einheit. Bei der Realisierung eines Prototyps wurden diese Funktionen verwendet und exemplarisch durch eine Handelsmöglichkeit ergänzt.

Das persönliche Fazit dieser Arbeit ist, dass eine Verschmelzung beider Spielkonzepte nur mit großer Kompromissbereitschaft bezüglich der Eigenschaften der Rollenspiele und jener der Pervasive Games möglich ist. Außerdem ist die Entwicklung eines Frameworks für das Design marktreifer Spielkonzepte aufgrund der unterschiedlichen Ausprägungen dieser Spiele als nicht sinnvoll zu erachten.

I. Glossar

Broadcast-Receiver *Android*-Komponente die als Nachrichtenempfänger zur Kommunkation zwischen Komponenten dient. Sie empfängt die Broadcast Intents die z.B. Informationen über den Netzwerkstatus geben können.

Intent Intents sind in *Android* Nachrichten zur Absichtserklärungen die zum Austauschen von Informationen zwischen Komponenten einer Anwendung oder von Anwendungen untereinander verwendet werden. Auf die Art und Weise können auch Activities gestartet werden.

Overlay Ein Overlay ist in *Android* wie eine durchsichtige Folie über der *Google Maps*™-Karte. Auf dieser kann gezeichnet und so der Karte weitere Informationen hinzugefügt werden.

Polling Als Polling bezeichnet man die zyklische Abfrage von Statusänderungen.

II. Verwendetes Kommunikationsprotokoll

In unterer Tabelle ist das auf dem CSV-Format basierende Kommunikationsprotokoll des Prototyps zwischen und Client und Server notiert. Die Nachrichten in der linken Spalte werden vom Client, die in der rechten vom Server versand. Innerhalb einer Tabellenzelle ist ein Dialog mit einer ggf. stattfindenden Antwort abgedruckt.

Client	**Server**
0;login;<username>;<password>	
	login;<userid> (userid= -1 wenn Nutzerdaten falsch)
<userid>;relogin	
	login;<userid> (userid = 0 wenn Relogin nicht mgl.)
<userid>;ownlocation;<latitude>;<longitude>;<altitude>	
	chardata;<charname>;<le>;<le_max>;<att>;<def>;<xp>;<level>;<coins>
<userid>;ordertraderdata;<buildingid>	
	traderdatastart
	traderdata;<itemid>;<typid>;<price> (0..*)
<userid>;buyitem;<itemid>	
	traderdatastart
	traderdata;<itemid>;<typid>;<price> (0..*)
<userid>;logout	
	ping
pong	

III. Eingetragene Warenzeichen

Apple Inc.:	Apple iPhone®;
Bethesda Softworks LLC:	The Elder Scrolls® Oblivion™ Fallout®
Blizzard Entertainment®:	Blizzard Entertainment® World of Warcraft™
Bluetooth SIG:	Bluetooth®
Commodore® Business Machines Inc.	PET
Deutsche Telekom AG:	T-Mobile
Eclipse Foundation Inc.:	Eclipse Eclipse Galileo
Electronic Arts Inc.:	Akalabeth™
European Telecom Standards Institute:	UMTS™
Fantasy Productions GmbH:	Das schwarze Auge: Drakensang
Funcom Inc.:	Anarchy Online™
Google Inc.:	Android Google™ ; Google Maps™; Open Handset Alliance™;
HTC Corporation:	HTC
Intel Corporation:	Intel® Intel® Core™ 2 Duo
Linus Torvalds:	Linux™
New England Biolabs:	GPS
Motorola Inc.:	Motorola

NCsoft Corporation:	Guild Wars® ArenaNet
Near Death Studios, Inc.:	Meridian59™
Origins Systems:	Ultima Online™
Peer Blue	Parallel Kingdom®
Pluto 13 GmbH:	Gothic™ Piranha Bytes
Samsung Electronics GmbH:	Samsung
Sony Online Entertainment LLC:	Everquest®
Sun Microsystems, Inc.:	J2SE™ Java™ Java EE™
Telecom Italia S.p.A.:	Telecom Italia
Ulisses Medien & Spiel Distributions GmbH:	Das Schwarze Auge (DSA)
Valve Corporation der Game Date Inc.:	Counter Strike™
Wikimedia Foundation, Inc.:	Wikipedia®
Wizards of the Coast LLC:	Advanced Dungeons & Dragons® Dungeons & Dragons®

IV. Abbildungsverzeichnis

V. Tabellenverzeichnis

VI. Literaturverzeichnis

[AchPieSim 2008] Leigh Achterbosch, Robyn Pierce, Gregory Simmons: *„Massively Multiplayer Online Role-Playing Games: The Past, Present, and Future"*, ACM: Computers in Entertainment, Oct 2007, Vol. 5/4, Article 9, New York: 2008.

[Android 2009] "Android Emulator", 2009, http://developer.android.com/intl/de/guide/developing/tools/emulator.html Zugriffsdatum: 24.12.2009

[AssTza 2006] Marios Assiotis, Velin Tzanov: *"A Distributed Architecture for MMORPG"*, Network and System Support for Games - Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games, SESSION: Scalability in MMOGs, Art.-Nr. 4, Singapore: 2006.

[Barton 2007a] Matt Barton: *„The History of Computer Role-Playing Games Part 1: The Early Years (1980-1983)"*, 2007, http://www.gamasutra.com/features/20070223a/barton_pfv.htm Zugriffsdatum: 22.12.09

[Barton 2007b] Matt Barton: *„The History of Computer Role-Playing Games Part 2: The Golden Age (1985-1993)"*, 2007, http://www.gamasutra.com/features/20070223b/barton_pfv.htm Zugriffsdatum: 22.12.09

[Barton 2007c] Matt Barton: *„The History of Computer Role-Playing Games Part 3: The Platinum and Modern Ages (1994-2004)"*, 2007, http://www.gamasutra.com/view/feature/1571/the_history_of_computer_.php?print=1 Zugriffsdatum: 22.12.09

[BecPan 2009] Arno Becker, Marcus Pant: *„Android – Grundlagen und Programmierung"*, dpunkt.verlag GmbH, Heidelberg: [1]2009

[Blizzard 2005] *World of Warcraft*™ Game Guide: *„Instanzierung"* http://www.wow-europe.com/de/info/basics/instancing.html Zugriffsdatum: 22.12.09

[Blizzard 2008] *„World of Warcraft™ zählt jetzt mehr als 11 Millionen Abonnenten weltweit“*
Pressemitteilung der Firma *Blizzard Entertainment®*, 2008
http://us.blizzard.com/de-de/company/press/pressreleases.html?081223
Zugriffsdatum: 21.12.09

[BjöHolLjuMan 2002] Staffan Björk, Jussi Holopainen, Peter Ljungstrand, Regan Mandryk:
„Special Issue on Ubiquitous Games“, Ubiquitous Computing, Vol. 6/ 6, 358-361, Springer-Verlag, London: 2002.

[Brockhaus 1992] „Schach“, in: *Brockhaus Enzyklopädie in 24 Bänden*, Band 19, Brockhaus, Mannheim: ^{18}1992.

[Chen 2004] Xin Chen:
„Developing Application Frameworks in .NET“, Apress: 2004.

[eGenesis 2009] *“A Tale in the Desert”*, 2009,
http://www.atitd.com
Zugriffsdatum: 22.12.09

[FdsA 2002] *„Fachlexikon der sozialen Arbeit“*, Verlag Soziale Theorie & Praxis, Frankfurt am Main: ^{5}2002.

[FriRitSch 2005] Tobias Fritsch, Hartmut Ritter, Jochen Schiller:
„The Effect of Latency and Network Limitations on MMORPGs – (A Field Study of Everquest® 2)”, in: NetGames '05: Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games, 2005, S. 1-9.

[GamHelJohVli 2007] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides:
„Entwurfsmuster“, Addison-Wesley, München: ^{3}2007.

[Google 2009] Google Inc.: *„Android Developers – The Developer’s Guide“*, 2009
http://developer.android.com/intl/de/guide/index.html
Zugriffsdatum : 22.12.09

[GriHal 2006] Carsten Griwodz, Pål Halvorsen:
„The Fun of using TCP for an MMORPG“, International Workshop on Network and Operating System Support for Digital Audio and Video Proceedings of the 2006 international workshop on Network and operating systems support for digital audio and video;
SESSION: Network Gaming, Art.-Nr. 1, ACM, New York: 2006.

[GuildWars 2005] ArenaNet: *„PvP“*, 2005
http://de.guildwars.com/pvp/
Zugriffsdatum: 22.12.09

[Hesselbach 2009] Martin Hesselbach:
“Android – Revolutioniert Google die Handywelt?”, 2009,
http://knol.google.com/k/android#
Zugriffsdatum: 22.12.09

[Jahnke 2009] Alex Jahnke:
„Gelebte Träume – Eine (Vor)Geschichte des LARP“
in: Dombrowski, Karsten (Hrsg.): LARP: Hinter den Kulissen. Aufsatzsammlung zum MittelPunkt 2009. Braunschweig: Zauberfeder, 2009.

[James 2009a] Derek James:
„Relativia Gameplay“, 2009
http://polyclefsoftware.blogspot.com/2009/07/relativia-gameplay.html
Zugriffsdatum: 22.12.09

[James 2009b] Derek James:
„Relativia is Submitted“, 2009
http://polyclefsoftware.blogspot.com/2009/08/relativia-is-submitted.html
Zugriffsdatum: 22.12.09

[Janus 2007a] Ulrich Janus:
„Was ist ein Fantasyrollenspiel?”
in: Ulrich Janus & Ludwig Janus (Hrsg.): „Abenteuer in anderen Welten“, Psychosozial-Verlag, Gießen: 2007, S. 25-26.

[Janus 2007b] Ulrich Janus:
„Wie laufen Rollenspiele ab?”
in: Ulrich Janus & Ludwig Janus (Hrsg.):
„Abenteuer in anderen Welten“, Psychosozial-Verlag, Gießen, 2007, S. 27-34.

[JegWib 2006] Kalle Jegers, Mikael Wiberg:
„Pervasive Gaming in the Everyday World”, Pervasive Computing, IEEE, Volume 5 /1, 2006, S. 78-85.

[Kent 2003] Steven L. Kent:
„Alternate Reality – The history of massively multiplayer online games“, 2003
http://archive.gamespy.com/amdmmog/week1/
Zugriffsdatum: 22.12.09

[LahRay 2006] Bernhard Lahres, Gregor Rayman:
„Praxisbuch Objektorientierung – Von den Grundlagen zur Umsetzung“, 2006
http://openbook.galileocomputing.de/oo/oo_03_prinzipien_005.htm
Zugriffsdatum 22.12.09

[LanHalNumLahMäyErm 2004] Petri Lankoski, Satu Haliö, Jani Nummela, Jussi Lahti, Frans Mäyrä, Laura Ermi:
„A Case Study in Pervasive Game Design: The Songs of North“, University of Tampere, Finland, 2004
ACM International Conference Proceeding Series,
Proceedings of the third Nordic conference on Human-computer interaction (NordiCHI ‘04), Vol. 82, S. 413-416.

[Leyde 2007] Irina Leyde:
„Rollenspiel online” in: Ulrich Janus & Ludwig Janus (Hrsg.):
„Abenteuer in anderen Welten“, Psychosozial-Verlag, Gießen, 2007, S. 149-176.

[Lüders 2008] Daniel Lüders:
„Netz-Teilchen – Erster Test des Google-Smartphones G1 mit Android-Betriebssystem“, c’t 24/08, Heise Zeitschriften Verlag, Hannover, S. 25-26.

[Mattern 2001] Friedemann Mattern:
„Pervasive Computing / Ubiquitous Computing“, ETH Zürich, 2001,
http://www.vs.inf.ethz.ch/publ/papers/UbiPvCSchlagwort.pdf
Zugriffsdatum: 22.12.09

[Meridian 2009] *“Accounting abgeschaltet”*, Pressemitteilung vom 06.11.2009
http://meridian59.de/
Zugriffsdatum: 22.12.09

[MönGriMid 2006] Christian Mönch, Gisle Grimen, Roger Midtstraum
„Protecting Online Games against Cheating“, Network and System Support for Games, Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games, Singapore: 2006.

[Morrill 2008] “Announcing the Android 1.0 SDK, release 1”, 2008,
http://android-developers.blogspot.com/2008/09/announcing-android-10-sdk-release-1.html
Zugriffsdatum: 22.12.09

[Nagel 2007] Rainer Nagel:
„Die Geschichte der Rollenspiele” in: Ulrich Janus & Ludwig Janus (Hrsg.): „Abenteuer in anderen Welten“, Gießen: 2007, S. 35-54.

[Napitupulu 2006] Jan Napitupulu:
„Entwicklung einer ortsabhängigen Spielesoftware für Pervasive Gaming“, Bachelorarbeit HAW Hamburg: 2006.

[OKS 2009] Abbildung:
„Pervasive Gaming related to other games“, OKS, TU-Berlin
http://www.oks.tu-berlin.de/forschung/technologien/pg/
Zugriffsdatum: 22.07.09

[PerBlue 2009] PerBlue Software:
“Parallel Kingdom®”
http://www.parallelkingdom.com/
Zugriffsdatum: 22.12.09

[Poitzmann 2007] Nikola Poitzmann:
„Sucht nach virtuellen Welten? – Aufbau und Wirkung des Online-Rollenspiels EverQuest®“, Tectum-Verlag, Marburg: 2007.

[Polyclefsoftware 2009] Polyclef Software LLC:
„Relativia“, 2009
http://www.polyclefsoftware.com/relativia.html
Zugriffsdatum: 22.12.09.

[Prinz 2006] Wolfgang Prinz:
„Pervasive Games“, 2006, Folie 11
http://www.iuk.fraunhofer.de/downloads/veranstaltungen/IuK-Pervasive%20Games-Prinz1.pdf
Zugriffsdatum: 22.12.09.

[Richtel 2009] Matt Richtel:
“*Google: Expect 18 Android Phones by Years’s End*“, New York Times, 27.5.2009
http://bits.blogs.nytimes.com/2009/05/27/google-expect-18-android-phones-by-years-end/
Zugriffsdatum: 22.12.09.

[RieWehFouNiePetCar 2007] Simon Rieche, Klaus Wehrle, Marc Fouquet, Heiko Niedermayer, Leo Petrak, Georg Carle:
“Peer-to-Peer-based Infrastructure Support for Massively Multiplayer Online Games”, 2007
http://www.net.in.tum.de/fileadmin/bibtex/publications/papers/ccnc2007.pdf
Zugriffsdatum: 22.12.09.

[Stäuble 2008] Markus Stäuble:
„Googles mobile Plattform Android“, 2008
http://www.heise.de/developer/artikel/Googles-mobile-Plattform-Android-227120.html?view=print
Zugriffsdatum: 28.2.09.

[Stolzenberg 2005] Justin Stolzenberg:
„Everquest® 2: PvP-Kampf im Sommer“, 2005
http://www.pcgames.de/aid,366021/Everquest-2-PvP-Kampf-im-Sommer/PC/News/
Zugriffsdatum: 22.12.09

[Wachholz 2005] Mark Wachholz:
„Die Geschichte des Schwarzen Auges – Ein historischer Überblick zum bekanntesten deutschen Rollenspiel” in: Momo Evers (Hrsg.): „Magische Zeiten”, Fantasy Productions, Ulm: 2005, S. 91-92.

[Weiser 1991] Mark Weiser:
„The Computer for the 21st Century”, Artikel, Scientific American, 1991
http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html
Zugriffsdatum: 22.12.09

[Wichter 2007] Andreas Wichter:
„Systeme, Welten, Abenteuer – eine Übersicht über die Welt des Rollenspiels” in: Ulrich Janus & Ludwig Janus (Hrsg.): „Abenteuer in anderen Welten“, Psychosozial-Verlag, Gießen: 2007, S. 55-75.

[Wikipedia® 2009a] „dnd“, 2009,
http://en.wikipedia.org/wiki/Dnd_(computer_game)
Zugriffsdatum: 24.12.09

[Wikipedia® 2009b] *„Landshuter Hochzeit“, 2009,*
http://de.wikipedia.org/wiki/Landshuter_Hochzeit
Zugriffsdatum 24.12.09

[ZhaKemDen 2008] Kaiwen Zhang, Bettina Kemme, Alexandre Denault:
„Persistence in Massively Multiplayer Online Games“, Network and System Support for Games, Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games, SESSION: Massively multiplayer online games, 2008, S. 53 – 58.

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit zum Thema

Konzeption und Realisierung eines Frameworks
für mobile Rollenspiele mit Android

im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

_____ _____

Ort, Datum Unterschrift des Studenten