Diplomarbeit im Fach Informatik

Annotationen im „Mobile Learning Game“

vorgelegt von

Christoph Wirz

geb. in Wangen bei Olten, Solothurn

Matrikelnummer 99-910-721

angefertigt am

Institut für Informatik

der Universität Zürich

bei

Prof. Dr. Gerhard Schwabe

unter Betreuung von

Christoph Göth

Abgabe der Arbeit: 15. März 2007

Zusammenfassung

Im Rahmen des Forschungsgebietes „Mobile Learning Games“ wurde eine interaktive und mobile Spielumgebung entwickelt, die es Studenten ermöglichen soll, die Universität besser kennen zu lernen.

In der vorliegenden Arbeit geht es darum, für das bestehende Mobile Game Handschrift-, Voice- und Foto-Annotationen zu implementieren. Da es in mobilen Umgebungen immer wieder zu Problemen mit dem zugrunde liegenden Netzwerk kommt und da es sich bei den Annotationen um Informationen mit grossen Datenmengen handelt, werden diese Umstände analysiert und entsprechende Lösungsvorschläge vorgestellt.

Das Resultat ist eine funktionsfähige Erweiterung des Mobile Games, welche durch einen Benutzertest erfolgreich getestet wurde.

Abstract

In the context of the research field „Mobile Learning Games“, there was developed a mobile game environment, which should help students to learn to know the university better.

This diploma thesis implements handwriting, voice and photo annotations for the existing Mobile Game application. Because there are always some problems in mobile environments with the underlying network and because these annotations contain a big amount of data, this paper discusses these circumstances and tries to show some solutions.

The result is a functional extension of the Mobile Game, which has been confirmed by a user test.

Inhaltsverzeichnis

1. Einleitung

Im Rahmen des Forschungsgebietes „Mobile Learning Games“ wurde durch Christoph Göth eine interaktive und mobile Spielumgebung entwickelt [Göth2003]. Diese Umgebung soll neuen Studenten[1] helfen, spielerisch die Universität kennen zu lernen. Inhalt dieses Spiels ist es, mit anderen Studenten zusammen Aufgaben zu wichtigen und interessanten Orten und Personen zu lösen. Dabei sollen sie durch mobile Kleingeräte unterstützt werden, die ihnen sowohl Navigation und Kommunikation ermöglichen als auch Information liefern. Auf dem mobilen Gerät sieht man die eigene aktuelle Position auf einer Karte. Diese Positionsinformationen ermöglichen es unter anderem, Anmerkungen (Annotationen) zu speziellen Orten zu erstellen.

Auf dieser Basis soll nun in dieser Diplomarbeit das Spiel um Handschrift-, Foto- und Voice-Annotationen erweitert werden. Implementiert waren bisher bereits normale Textannotationen.

Die Arbeit besteht aus drei Teilbereichen. Im ersten Teil werden die drei Annotationsarten kurz erläutert und auf bestehende Projekte eingegangen und diese miteinander verglichen. Danach werden die Eigenschaften von drahtlosen Netzwerken angeschaut und diese im Zusammenhang mit dem Mobile Game und den Annotationen analysiert.

Im zweiten Teil wird die Implementierung beschrieben, sowie besondere Entwicklungsprozesse näher erläutert.

Im dritten und letzten Teil wird der Benutzertest vorgestellt und dessen Ergebnisse analysiert. Im Fazit werden persönliche Eindrücke betreffend des Mobile Games und der Implementierung erläutert und auf mögliche zukünftige Entwicklungen hingewiesen.

[1] Die maskuline Form wird in der ganzen Arbeit stellvertretend für beide Geschlechter verwendet

2. Grundlagen

Dieses Kapitel behandelt die theoretischen Grundlagen von Annotationen und analysiert bestehende Projekte, die in diesem Bereich bereits entwickelt wurden.

2.1. Annotation

Eine Annotation ist eine Anmerkung zu einem Objekt. Sie besteht aus einem Anker, dem Annotationsinhalt und der Verknüpfung zwischen beiden [FroSchw2004]. Der Anker bestimmt, worauf sich die Annotation bezieht. Der Annotationsinhalt kann sowohl Text als auch Bilder oder andere Daten enthalten. Laut Bernheim Brush [BeBr2002] kann jeder Medientyp annotiert werden.

Annotationen erlauben es, Gedanken und Überlegungen zu einem bestimmten Objekt festzuhalten. Dabei ist der Kontext der Annotation jederzeit durch seinen Anker bestimmt und klar definiert. Die Annotation ermöglicht sowohl die Bezugnahme auf Materialien als auch auf andere Annotationen [Kienle2003].

2.2. Annotation im Mobile Game

Im Mobile Game können Annotationen an beliebigen Stellen im Spiel erstellt werden. Sie sind auf der Spielkarte durch spezielle Symbole gekennzeichnet und können von jedem Spieler betrachtet und wiederum annotiert werden.

Aufgabe in dieser Arbeit ist, die bereits im Mobile Game implementierte Text-Annotation zu erweitern und Foto-, Voice- und Handschrift-Annotationen hinzuzufügen. Eine solche Anmerkung im Mobile Game besteht aus einem Titel, einem Erstellernamen, den Positionsinformationen sowie dem eigentlichen Inhalt.

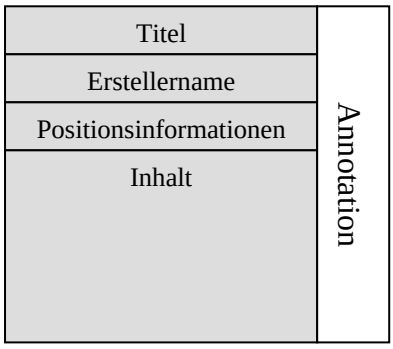


Abbildung 1 Aufbau einer Annotation im Mobile Game

Die Annotationen sollen dann an einen zentralen Server gesendet und dort verwaltet werden. Für andere Spielteilnehmer soll, sofern dies in den Regeln erlaubt wurde, eine Annotation ersichtlich sein, sobald sie erfolgreich an den Server versendet wurde.

2.2.1. Handschrift-Annotation

Bei einer Handschrift-Annotation handelt es sich um eine Annotation, die als Zeichnung dargestellt wird. Dabei soll es dem Benutzer mit dem Eingabestift möglich sein, sowohl handgeschriebene Texte als auch Zeichnungen zu erstellen. Dieser Eingabestift wird bei den meisten mobilen Kleingeräten als Eingabehilfe verwendet und ist zentral für die Benutzeroberfläche.
Diese Annotationsart soll vor allem dann zum Einsatz kommen, wenn es sinnvoll ist, kleine Schemas abzuzeichnen oder kurze Notizen zu machen.

2.2.2. Voice-Annotation

Eine Voice-Annotation ist eine Sprachnachricht, die mit einem im mobilen Endgerät eingebauten Mikrofon aufgezeichnet werden kann. Das Schreiben mit dem Eingabestift empfinden viele Personen als mühsam und ineffizient. Mit Hilfe der Voice-Annotationen können diese Personen umfangreiche Annotationen erstellen, ohne auf die Texteingabe angewiesen zu sein.

2.2.3. Foto-Annotation

Eine Foto-Annotation ist ähnlich wie eine Handschrift-Annotation. Sie setzt ebenfalls auf die visuelle Information. Sie wird jedoch mittels der integrierten Kameras der mobilen Kleingeräte erstellt. Somit soll es den Spielern des Mobile Games möglich sein, an beliebigen Stellen Fotos zu erstellen und diese dann den Mitspielern zur Verfügung zu stellen.

2.3. Analyse bisheriger Projekte

In diesem Kapitel werden verschiedene Projekte vorgestellt, welche ebenfalls Annotationen verwenden. Ziel ist, diese Projekte zu analysieren und deren Stärken und Schwächen kennen zulernen, um Erfahrungen für die eigene Implementierung zu erlangen.

2.3.1. Flickr

Bei Flickr [Flickr] handelt es sich um ein Online-Portal, welches Benutzern ermöglicht, Fotos online zu verwalten. Man kann auf diesem Portal seine eigenen Fotos ablegen und diese für andere Benutzer zugänglich machen. Dabei können auch andere Personen die eigenen Fotos kommentieren.

Flickr bietet die Möglichkeit, so genannte „Notes and Tags“ zu jedem Bild hinzuzufügen. Bei diesen Notes and Tags handelt es sich um Metainformationen, welche die Verständlichkeit des Bildes erhöhen. Notes sind Bereiche im Bild, die dann ersichtlich werden, wenn ein Betrachter mit der Maus über das Bild fährt. Dann werden diese Regionen hervorgehoben und eine Notiz zum entsprechenden Bereich ausgewählt. Somit ist es möglich, einzelne Bildteile zu annotieren und auf wichtige oder besonders interessante Details hinzuweisen. In Abbildung 2 sieht man zwei „Notes“ die im Bild definiert sind.

Abbildung 2 „Notes“ in Flickr

Durch Tags kann man die Bilder einzelnen Kategorien zuweisen. Durch diese Kategorisierung ist es einfacher, Bilder zu finden.

Flickr bietet weiter die Möglichkeit, den Fotos Informationen über ihre Herkunft zuzuweisen. In Flickr wird das mit „Geotag“ bezeichnet.

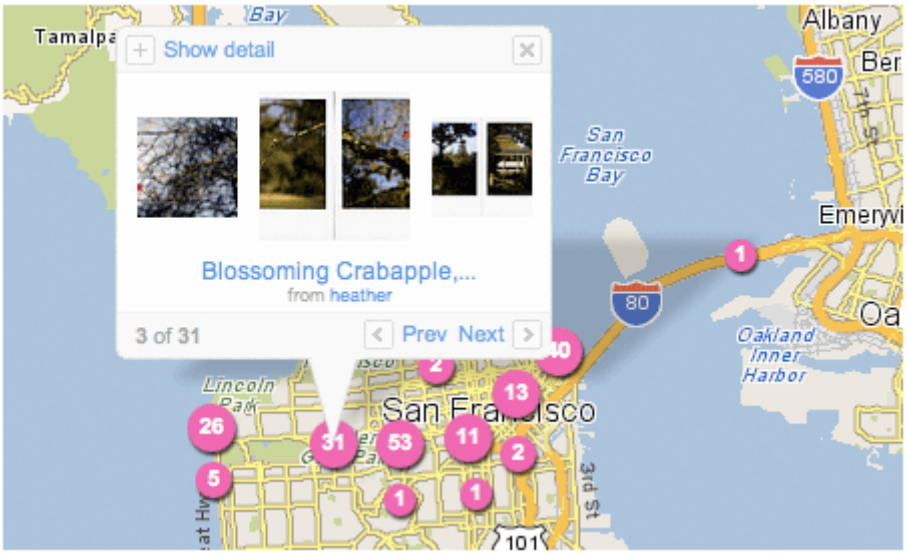


Abbildung 3 „Geotag“ in Flickr

Abbildung 3 zeigt eine Karte, die mit einzelnen Punkten markiert ist. Die Anzahl der Fotos an der entsprechenden Stelle ist im Punkt als Zahl angegeben. Durch einen Klick auf einen solchen Punkt werden die Fotos angezeigt.

2.3.2. GeoNotes

Mit GeoNotes kann man die auf einer Karte abgebildeten realen Orte mit virtuellen Notizen versehen. Diese Notizen können dann von anderen Personen betrachtet werden. GeoNotes ist in Java implementiert und kann sowohl mit mobilen Kleingeräten als auch auf Desktop Computersystemen eingesetzt werden.

Ziel von GeoNotes ist, dass solche virtuellen Notizen einfach und schnell erstellt werden können. Dabei muss der Ersteller einer solchen Notiz folgende vier Komponenten spezifizieren [Espinoza]:

- **Titel**

  Im GeoNote System erscheint der Titel als Kopfzeile in Listen. Er dient als Beschreibung für den Inhalt.

- **Empfänger**

  GeoNote ermöglicht es, Notizen sowohl einzelnen Empfängern als auch ganzen Gruppen oder allen Personen zuzuweisen. Damit kann der Zugriff auf Inhalte beschränkt werden.

- **Unterschrift**

  Mit Hilfe der Unterschrift kann der Notiz einen Namen zugewiesen werden. Ziel

ist es nicht, die Identität des Benutzers zu kennen, sondern den Ersteller auf Grund seines Pseudonyms wieder zu erkennen.

- **Ort**

  Es besteht die Möglichkeit, einen Ort für die Notiz anzugeben. Dies ist dann sinnvoll, wenn die Positionsinformationen ungenau sind.

In Abbildung 4 sieht man eine Übersicht von GeoNotes. Einer Notiz kann man einen Ort hinterlegen.

Abbildung 4 GeoNotes [Pearsson2002]

Abbildung 5 zeigt eine einfache Notiz. Diese kann durch andere Personen mit Kommentaren versehen werden.

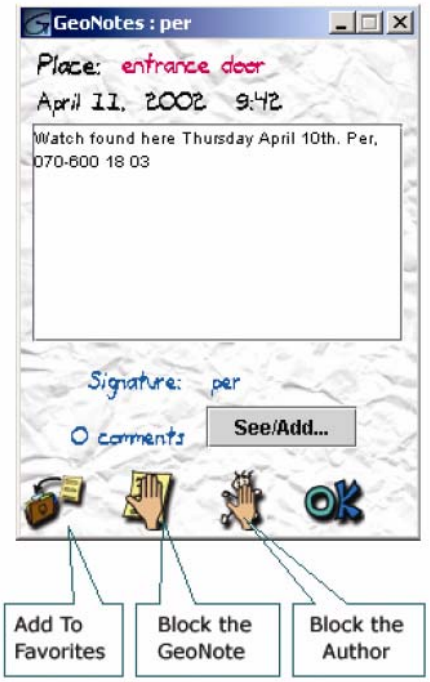


Abbildung 5 GeoNotes – Anzeigen einer Notiz [Pearsson2002]

2.3.3. CatchBob!

CatchBob! ist eine experimentelle Plattform in Form eines mobilen Spiels, das für psychologische Tests verwendet wird [CatchBob]. Ziel ist, dass die Spieler ein virtuelles Objekt finden, welches sich im Spielbereich befindet. Um das Spiel zu gewinnen, müssen drei Gruppen das Objekt in der realen Welt so umstellen, dass der Abstand jeder Person zum Objekt unter einen bestimmten Wert fällt. Das Spiel gilt dann als gewonnen, wenn sich die Spieler in Form eines Dreiecks um das Objekt platziert haben. Jedes Team sieht die Partner-Teams auf der Spielkarte. Dem Spieler wird mittels einer Distanzanzeige vermittelt, wie weit entfernt er sich vom gesuchten Objekt befindet. Den Spielern werden einfache Kommunikationsmittel zur Verfügung gestellt. Unter anderem kann man auf einen Partner (auf der Spielkarte) klicken und dann einen Strich in die Richtung ziehen, in welche sich die andere Person bewegen soll. Zudem kann man Meldungen an alle anderen Spieler versenden.

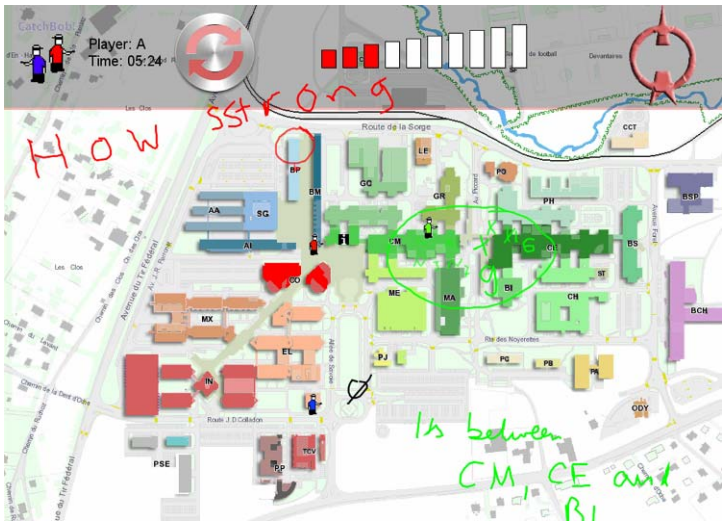


Abbildung 6 Spielansicht bei CatchBob!

In der Abbildung 6 sieht man die Spielansicht bei CatchBob! aus Sicht eines Spielers. Oben in der Mitte der Ansicht befindet sich der Indikator für die Distanz zum gesuchten Objekt.

CatchBob! wurde unter anderem in der Arbeit “A Mobile Game to Explore the Use of Location Awareness on Collaboration” [Nova2005] als Spielumgebung verwendet.

2.3.4. HyConExplorer

Bei HyConExplorer handelt es sich um eine Applikation, die den Benutzer dabei unterstützt, die reale Welt zu durchsuchen und zu annotieren. Der HyConExplorer läuft auf Tablet PC und Smartphones, die J2ME (Java 2 Micro Edition) unterstützen. Mit Hilfe des HyConExplorers lassen sich Orte an bestimmten Stellen auf einer Karte erstellen. In Abbildung 7 sieht man die Hauptansicht der Applikation. Im „Object browser“ befinden sich alle erfassten Orte. Diese werden in der „Map view“ mit blauen Punkten gekennzeichnet. Der rote Kreis markiert die eigene Position.

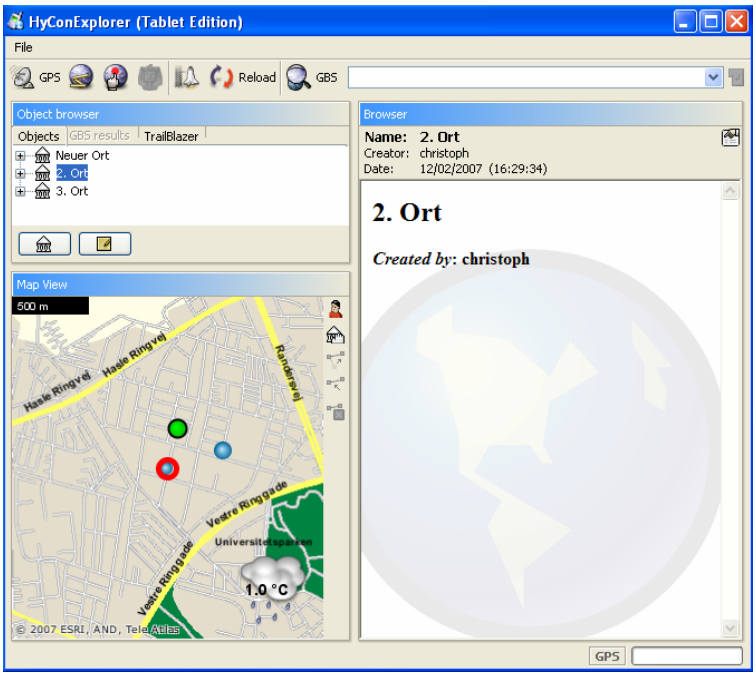


Abbildung 7 HyConExplorer – Hauptansicht

Diese erfassten Orte können mit einer Annotation versehen werden, welche aus Text oder aus anderen Mediendaten, wie zum Beispiel Bilder oder sogar Videos bestehen. In Abbildung 8 sieht man den Dialog zur Erfassung einer Annotation zu einem bestimmten Ort.

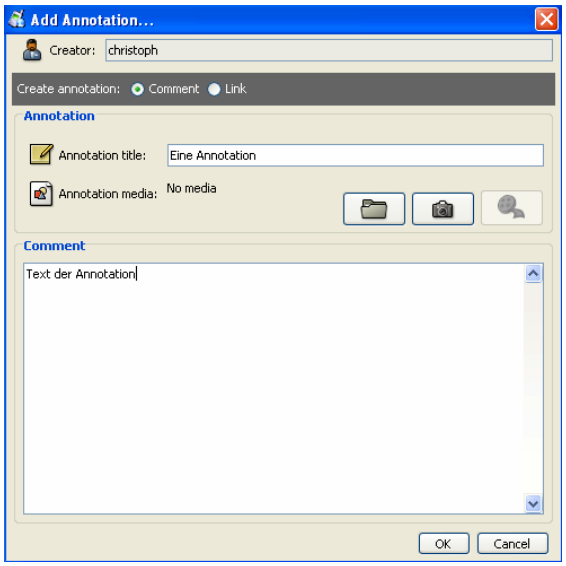


Abbildung 8 HyConExplorer – Erstellen einer Annotation

Jede Annotation kann selbst wieder annotiert werden. Dies zeigt Abbildung 9 am Beispiel von Ort 2.

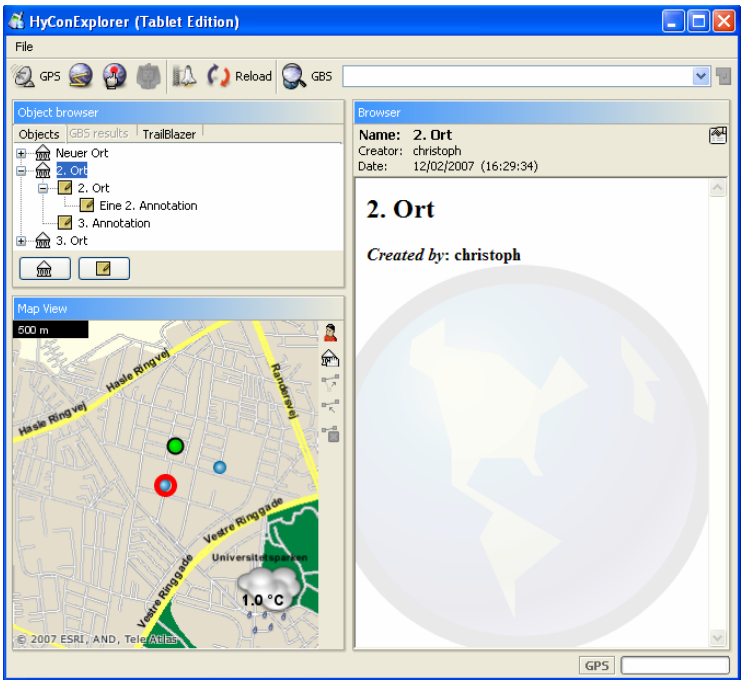


Abbildung 9 HyConExplorer – Annotieren einer Annotation

2.3.5. Vergleich

Mit Ausnahme von Flickr arbeiten alle betrachteten Systeme mit aktuellen Positionsinformationen, die durch verschiedene Ortungssysteme erlangt werden (GPS, WLAN etc). Bei Flickr muss der Benutzer selber entscheiden, zu welchem Ort das Foto gehört. Er muss es also von Hand zuweisen und wird dabei nicht durch das System unterstützt. Alle untersuchten Projekte besitzen die Möglichkeit, Orte oder Objekte zu annotieren. In Flickr wird dies zum Beispiel durch das Einfügen von Kommentaren zu einem Bild ermöglicht. Bei CatchBob! kann man auf der ganzen Karte Notizen und Zeichnungen erstellen. Bei GeoNotes und HyConExplorer werden die Annotationen expliziter behandelt und sind von zentralerer Bedeutung. GeoNotes verwendet dazu so genannte virtuelle Notizen, um zusätzliche Informationen zu Orten zu liefern. Diese Notizen können wiederum mit Kommentaren versehen werden. Beim HyConExplorer sind die Annotationen klar ersichtlich. Jeder Ort kann annotiert werden. HyConExplorer ist das einzige System, welches die volle Unterstützung für verschiedene Medientypen bietet.

Das Mobile Game ist derzeit aufgrund seiner Funktionalitäten zwischen GeoNotes und dem HyCon-Explorer anzusiedeln. Es bietet ebenfalls die Möglichkeit von Annotationen und nutzt ebenfalls Positionsinformationen, um diese bestimmten Orten zuzuweisen.

3. Probleme der Realisierung

Die Arbeit „Photo Annotation on a Camera Phone“ [Wilhelm2004] beschreibt ein System, welches es Benutzern erlaubt, digitale Fotos zu annotieren. Dazu wurden mobile Telefone mit integrierten Kameras verwendet. Die damit erstellten Fotos konnten über ein drahtloses GSM / GPRS-Netz an einen Server versendet werden. Während eines Tests äusserte sich ein Benutzer (während dem Übertragen eines Fotos an den Server) wie folgt:

„I have to keep staring at the screen to check for change even though would rather pay attention to other things around me.“ [Wilhelm2004 s.1404]

Diese Aussage zeigt, dass Verzögerungen und Unterbrüche in drahtlosen Netzwerken zu Verunsicherungen der Benutzer führen kann. Die gleichen Schwierigkeiten werden auch in der Arbeit „Metadata Creation System for Mobile Images“ [Sarvas2004] beschrieben.

Dieses Kapitel untersucht diese Probleme und stellt entsprechende Lösungen vor.

3.1. Netzwerke

Der wichtigste Punkt bei einer Applikation mit mehreren Clients auf verschiedenen Systemen ist der Austausch von Nachrichten über ein Medium.

Im Mobile Game kommen ausschliesslich mobile Endgeräte zum Einsatz. Deshalb ist es nahe liegend, diese Netzwerkumgebung genauer zu untersuchen.

3.1.1. Eigenschaften von Netzwerken des Standards 802.11

Der IEEE-Standard 802.11 beschreibt die derzeit am weitesten verbreitete Gruppe von drahtlosen Netzwerken. Dieser Standard gehört in den Bereich der 802.x Mitglieder, zu welchen auch Ethernet (802.3) und Token Ring (802.5) gehören [Schiller2003].

3.1.2. Dienstgüte

Drahtlose Netzwerke sind in praktisch allen Dienstgüteparametern schlechter als leitungsgebundene Netzwerke. Dazu zählen Parameter wie Bandbreite, Übertragungsfehlerraten, Verzögerungen und Verzögerungsschwankungen [Schiller2003]. Die Bandbreite liegt beispielsweise bei heutigen drahtgebundenen Netzwerken typischerweise bei 100-1000 MBit/s. Die Nutzdatenrate von drahtlosen LAN hingegen liegt im Moment (je

nach Signalqualität) bei 1-10 MBit/s. Ähnlich sieht es bei der Anzahl der Übertragungsfehler aus. Diese liegen bei drahtgebundenen LAN in der Grössenordnung von 10^{-12} im Vergleich zu 10^{-4} beim drahtlosen Netzwerk. Die höheren Fehlerraten führen automatisch auch zu mehr Verzögerungen und Verzögerungsschwankungen.

3.1.3. Bewegliche Clients

Eine weitere Eigenschaft von drahtlosen Netzen ist, dass es bewegliche Clients geben kann. Dies ist insbesondere im Zusammenhang mit dem Mobile Game von grosser Bedeutung, da sich die Spieler in ständiger Bewegung befinden.
Die Bewegung führt zu verschiedenen Effekten, die meistens mit Verlangsamung oder sogar Unterbrechung der Verbindung zusammen hängen.

Wechsel von Zugriffspunkten

Dieser Effekt tritt auf, wenn sich ein Client aus dem Wirkungsbereich einer Antenne beziehungsweise eines Zugriffspunktes in den Wirkungsbereich eines anderen bewegt. Dies kann zu Verzögerungen von versendeten Paketen oder sogar zu Paketverlusten führen.

Änderung der Signalqualität

Durch die Bewegung ändert sich die Qualität des übertragenen Signals ständig. Dies führt zu Unregelmässigkeiten im Paketverkehr. So kann ein Paket schnell versendet werden, ein anderes hingegen benötigt dann mehr Zeit oder erreicht den Empfänger gar nicht mehr.

Verbindungsunterbrüche wegen mangelnder Netzabdeckung

Die Netzwerkabdeckung kann nicht überall gleichermassen gewährleistet werden. Es ist also sehr gut möglich, dass nicht überall eine Verbindung besteht oder diese für eine gewisse Zeit unterbrochen wird.

3.2. Probleme in Zusammenhang mit Annotationen

Dieses Kapitel betrachtet die Eigenschaften der drahtlosen Kommunikation in Zusammenhang mit Foto-, Voice- und Handschrift-Annotationen.

3.2.1. Technische Probleme

Die Annotationen, die diese Arbeit behandelt, haben alle eine gemeinsame Eigenschaft. Ihr Inhalt besteht in der Regel aus einer grösseren Datenmenge. In der Datenbanktechnik spricht man von so genannten BLOBs[2]. Deshalb wurde entschieden, die Annotationen in der Implementierung als BLOB-Annotationen zu bezeichnen. Aufgrund dieser grossen Datenmengen entstehen beim Übertragen von solchen Inhalten in drahtlosen Netzwerken Probleme.
Gerade Übertragungsunterbrüche und Verzögerungen werden grosse Auswirkungen auf die Stabilität und Funktionalität des Systems haben.

3.2.2. Benutzerspezifische Probleme

Die grösseren Fehlerübertragungsraten und Verzögerungen haben nicht nur technische Auswirkungen auf die Applikation sondern auch Einfluss auf den Benutzer selbst. Durch die Unsicherheiten auf der Netzwerkseite wird die Interaktion mit dem Benutzer beeinflusst und muss von Anfang an in der Entwicklung in Betracht gezogen werden. So wird es zum Beispiel von grosser Bedeutung sein, wie man mit den Verzögerungszeiten umgeht. Wenn zum Beispiel ein Benutzer eine gewisse Datenmenge an den Server übermitteln will, ist es wichtig, dem Benutzer mitzuteilen, wie schnell es vorwärts geht. Ohne Benachrichtigung über den Status der Datenübertragung wird der Benutzer mit zunehmender Wartezeit ungeduldig und vermutet eventuell sogar ein Problem mit der Applikation. In diesem Falle wäre dann eine Fortschrittsanzeige angebracht. Dabei ist es weniger wichtig, wie schnell sich die Daten übertragen lassen, sondern viel mehr, dass sich die Daten übertragen lassen. Die Interaktion und Benachrichtigung ist also dementsprechend wichtig.

[2] Binary Large Object

3.3. Lösungsansätze

Um mit den Problemen von drahtlosen Netzwerken besser umgehen zu können, kann man einige Überlegungen anstellen. Für die meisten Probleme gibt es allerdings aus der Applikationssicht keine Lösungen. Man kann lediglich die Auswirkungen der Probleme reduzieren, wenn man sie bereits zur Entwurfszeit berücksichtig. Deshalb werden in den folgenden Kapiteln genau diese Eigenschaften und Lösungsansätze genauer betrachtet.

3.3.1. Datenkomprimierung

Ein wichtiger Punkt bei der Übermittlung von Daten ist die Datengrösse. Es ist sinnvoll, diese so klein wie möglich zu halten, also zu komprimieren. Das erscheint auf den ersten Blick einfach und ist relativ schnell implementiert. Trotzdem muss man bedenken, dass die Prozessoren auf den mobilen Clients um ein vielfaches leistungsschwächer sind als solche von herkömmlichen Desktop Computern. Dies führt bei grossen Datenmengen, welche komprimiert und dekomprimiert werden müssen, zu grösseren Wartezeiten. Es gilt also, einen geeigneten Mittelwert zwischen der Komprimierungsrate und der zur Datenaufbereitung verwendeten Rechenzeit zu finden.

3.3.2. Aufteilung der Daten

Da es bei drahtlosen Netzwerken immer wieder zu Verzögerungen oder Unterbrüchen kommt, ist es sinnvoll, die Daten in kleinere Pakete aufzuteilen und einzeln zu versenden. Dies hat den Vorteil, dass man eine bessere Überwachung des Transfervorganges hat, als wenn man nur ein grosses Paket übermittelt. Folgende Grafik veranschaulicht den Unterschied der beiden Pakete, wenn davon ausgegangen wird, dass die Gesamtdatengrösse bei beiden gleich ist.

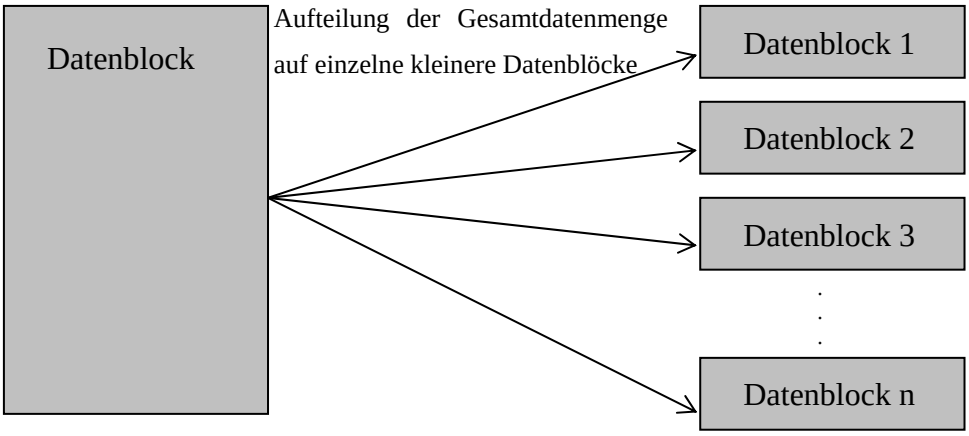


Abbildung 10 Datenpakete

Jeder Teildatenblock verfügt über die gleiche Datengrösse. Ausnahme bildet lediglich der letzte Block da dieser die Restdatenmenge beinhaltet.

Eine Analyse des Transfervorganges veranschaulicht den Vorteil der Methode mit den kleineren Datenpaketen. Wir gehen nun davon aus, dass wir eine grosse Datenmenge über das Netzwerk übermitteln müssen. Wenn man nur ein Datenpaket verwendet, werden die gesamten Daten gekapselt und über das Netzwerk versendet. Das Problem bei dieser Vorgehensweise ist, dass man die Kontrolle über das gesendete Paket verliert. Man kann nur warten und den Vorgang bei einem Versagen der Übermittlung neu starten oder eine entsprechende Fehlermeldung generieren. Es ist nicht möglich, Aussagen über den Status des Datenpakets zu machen.

Das Prinzip bei kleineren Paketen ist genau gleich wie bei einem grossen Paket. Man hat aber mehr Informationen über den Gesamtstatus. Man weiss, wie viel Daten bereits übermittelt wurden und wie viele noch ausstehen. Zusätzlich ist bekannt, welches Paket gegebenenfalls nicht übermittelt werden konnte.

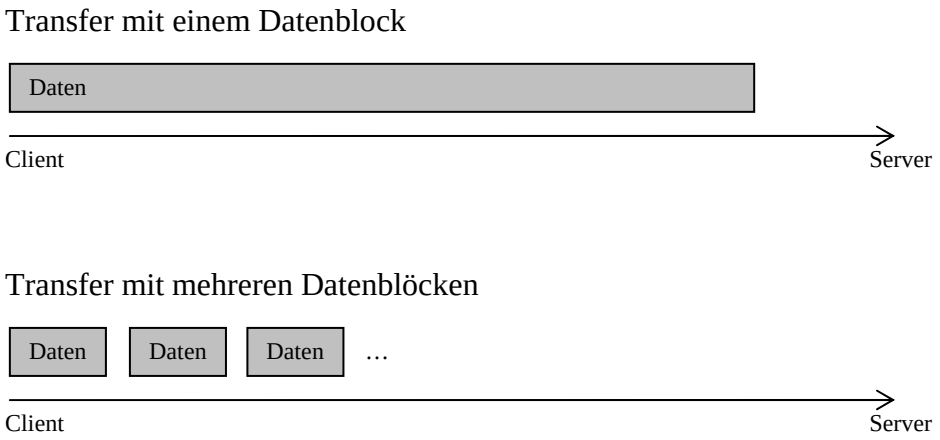


Abbildung 11 Transfervorgang mit verschiedenen Datenpaketen

Natürlich ist die Wahl der Datengrösse der Teilblöcke für die Aussagekraft des Übermittlungsstatus relevant. Zudem muss auch bedacht werden, dass die Aufteilung in kleinere Einheiten mit Mehraufwand auf der Transportschicht des Netzwerkes verbunden ist. Denn es wird pro Übertragungseinheit eine neue TCP Verbindung hergestellt.

Das TCP Paket kann theoretisch 65’495 Datenbyte enthalten [Tanenbaum2003].

Es ist deshalb sinnvoll, diese Grenze nach oben nicht zu überschreiten, da sonst mehrere TCP Pakete pro Datenblock über das Netz versendet werden müssen und dies zu einer Mehrbelastung des Netzwerkes führt.
Um die optimale Paketgrösse zu ermitteln, wurden verschiedene Tests durchgeführt, welche die durchschnittliche Grösse der einzelnen Annotationen ermitteln sollten.

Annotation	Durchschnittliche Grösse in KB (Kilo Byte)
Handschrift-Annotation	5-10
Voice-Annotation	20-100
Foto-Annotation	30-60

Tabelle 1 Grösse von Annotationen

Tabelle 1 zeigt die gemessenen Grössen der Annotationen, wie sie im Mobile Game verwendet werden. Bis auf die Voice-Annotation (hier ist die Grösse abhängig von der Dauer der Aufzeichnung) variieren die Grössen nicht stark. In der Praxis wird oftmals eine Grösse von 1'460 Datenbytes gewählt, um in einen einzelnen Ethernet-Rahmen inklusive IP- und TCP-Header zu passen [Tanenbaum2003]. Ein Ethernetrahmen kann 1'500 Bytes an Daten beinhalten. In dieses Datensegment gehören sowohl der IP-Header als auch der TCP-Header der übergeordneten Protokolle. TCP- und IP-Header haben jeweils eine feste Grösse von 20 Bytes. Daraus resultiert die effektive Grösse für die Nutzdaten von 1'460 Bytes.

3.3.3. Verteilung von eindeutigen Kennnummern

Die Aufteilung der Daten einer Annotation führt zu einem weiteren Problem auf der Seite des Servers. Dieser muss bei jedem ankommenden Datenpaket wissen, zu welcher Annotation dieses gehört. Das bedeutet, dass jede Übertragungseinheit eine eindeutige Kennnummer aufweisen muss. Und das wiederum bedeutet, dass jede Annotation eine eindeutige Nummer besitzen muss. Diese Nummer muss global eindeutig sein. Das bedeutet, dass es im ganzen System[3] keine zweite gleiche Kennnummer geben darf. Es ist daher nicht möglich, eine solche Nummer vom Client vergeben zu lassen, da dieser

[3] System bedeutet hier Server und alle Clients

keine Kenntnis über die bereits vergebenen und im System existierenden Nummern hat[4].

Dies führt dazu, dass der Server die Kennnummern verwalten und sie entsprechend an die Clients verteilen muss, wenn er dazu aufgefordert wird.

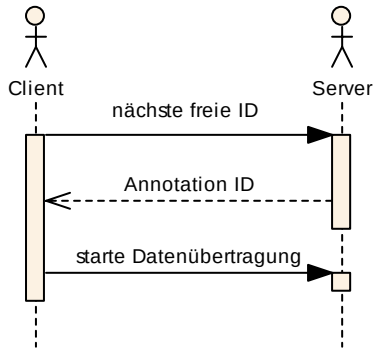


Abbildung 12 Anforderung einer eindeutigen Annotationsnummer

In Abbildung 12 sieht man, wie eine solche Kennummeranfrage an den Server aussieht. Der Server hält dabei jeweils die höchste vergebene Nummer und erhöht diese bei einer Anfrage um eins. Somit ist sichergestellt, dass jede Annotation über ihre ID angesprochen werden kann und dass es keine Duplikate im System gibt.

3.3.4. Bestätigungsmeldungen auf Applikationsebene

Um über das Netzwerk kommunizieren zu können, verfügen Sender und Empfänger über einen Puffer, in dem die ankommenden Daten zwischengespeichert werden. In den Puffer werden Daten geladen und verweilen dort, bis sie von der entsprechenden Applikation abgerufen werden. Die Anwendung hat jedoch keine direkten Einflussmöglichkeiten, um diesen Puffer zu steuern oder zu verwalten.

Bei der Übertragung der Daten kommt es immer wieder zu Verzögerungen oder anderen Übertragungsfehlern und führt dazu, dass die Pakete mehrfach übermittelt werden müssen. Bei mehreren Clients kann es durchaus auch sein, dass viele Daten gleichzeitig von verschiedenen Quellen eintreffen. Normalerweise stellt der Puffer an sich keinen Engpass für das System dar. Dieser sollte auch mit grossen Datenmengen von vielen verschiedenen Clients zurechtkommen. Kritisch wird es aber dann, wenn die Bear-

[4] Die Clients kommunizieren nicht untereinander.

beitung der aus dem Puffer ausgelesenen Daten zeitaufwändig und rechenintensiv ist. Dann kann es theoretisch dazu führen, dass der Puffer immer weiter anwächst, und die Applikation die Daten nicht schnell genug abarbeiten kann. Um eine Überflutung in diesem Sinne zu verhindern, werden in diesem Kapitel ein paar Massnahmen vorgestellt.

Wenn aus Sicht der Applikation verhindert werden soll, dass diese mit zu vielen Daten überhäuft wird, die sie nicht mehr behandeln kann, dann braucht es eine Art Bremsmechanismus, der verhindert, dass Daten ständig eintreffen. Eine Möglichkeit, dies zu erreichen, sind so genannte Bestätigungsmeldungen. Dazu wird jedes einkommende Datenpaket „bestätigt“, in dem der Empfänger dem Sender mitteilt, dass das Paket erfolgreich empfangen wurde und dass er das nächste senden darf. Abbildung 13 verdeutlicht einen solchen Ablauf.

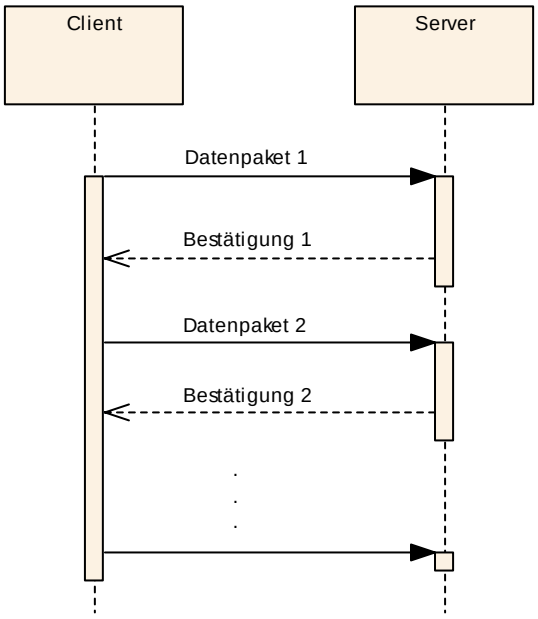


Abbildung 13 Datenübertragung mit Bestätigungsmeldungen

Der Client beginnt mit der Datenübertragung und übermittelt sein erstes Datenpaket an den Server. Er wartet dann, bis der Empfänger dieses quittiert hat. Die Quittierung erfolgt durch so genannte ACK-Meldungen[5]. Erst wenn der Sender die entsprechende Meldung erhalten hat, versendet er sein zweites Datenpaket. Dies führt dazu, dass der Server immer nur so viele Daten erhält, wie er auch tatsächlich abarbeiten kann.

[5] ACK steht für acknowledge (dt. Bestätigung)

3.3.5. Verteilung der Annotationen an Clients

Sobald die Daten für eine Annotation vollständig vom Server empfangen wurden, ist die Annotation bereit, um an alle beteiligten Clients versendet zu werden. Da es aber nicht sinnvoll ist, die gesamte Annotation inklusive ihrer Daten zu übermitteln, wird nur ein kleiner Teil übertragen. Dazu gehört neben Titel und Kennnummer auch die Information über die Datengrösse. Ein Client hat dann die Möglichkeit, mittels der Kennnummer die Daten der Annotation vom Server zu erhalten, indem er eine Request-Meldung an den Server sendet.

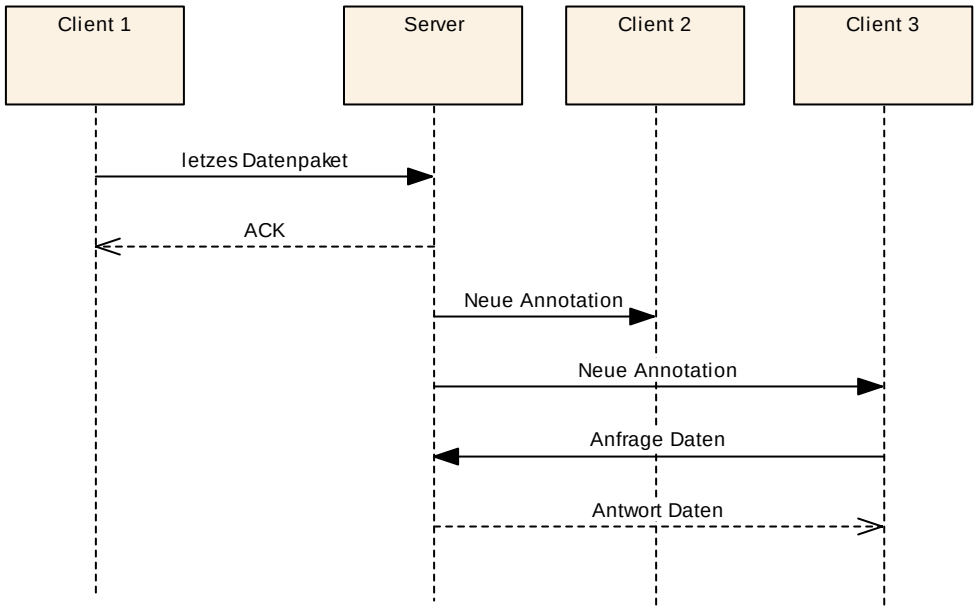


Abbildung 14 Ablauf Verteilung einer Annotation

In Abbildung 14 ist ersichtlich, wie eine solche Verteilung von Annotationen an mehrere Clients aussieht. Sobald Client 1 seine Datenübermittlung beendet hat, steht auf dem Server eine neue vollständige Annotation zur Verfügung. Der Server benachrichtigt alle Clients und übermittelt ihnen die Informationen zu Titel, Annotationskennummer und Datengrösse. Client 3 interessiert sich für die Annotation und startet eine Anfrage an den Server. Diese Anfrage beinhaltet wiederum die Annotationsnummer sowie den Startpunkt der Daten[6] und die Datenmenge, die er erhalten will.

[6] Startoffset

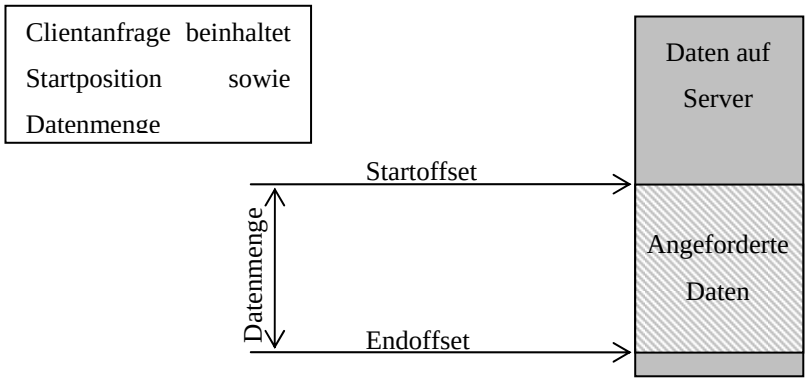


Abbildung 15 Datenblockanfrage

Abbildung 15 zeigt das Schema einer solchen Datenabfrage. Die Logik ist in diesem Fall auf den Client ausgelagert. Der Client muss wissen, welche Daten er will und welche er bereits hat.

4. Implementierung

Es wurde schon relativ früh mit der Implementierung begonnen, da es für diese Arbeit von grosser Wichtigkeit war, mit verschiedenen Unbekannten umzugehen. Dazu zählten unter anderem die Fragen, ob es möglich wäre, mit Java die integrierte Kamera anzusteuern oder wie man die Sprachaufzeichnungskomponente der PDAs nutzen kann? Literatur war in diesen Bereichen leider nicht sehr zahlreich verfügbar. Einerseits weil es sehr spezifische Probleme sind und andererseits weil es sich dabei um ein noch relativ junges Entwicklungsgebiet handelt.

Die zum Zeitpunkt dieser Arbeit vorliegende Version des Mobile Games beruht auf der Architektur von Denis Brunner, die er im Zusammenhang mit seiner Diplomarbeit [Brunner] entwickelt hatte. Daniel Suter analysierte in seiner Diplomarbeit [Suter] unter anderem den Aufbau der System Architektur und stellte fest, dass sowohl der Mobile Game Client als auch der Mobile Game Server aus einer dreischichtigen Architektur besteht.

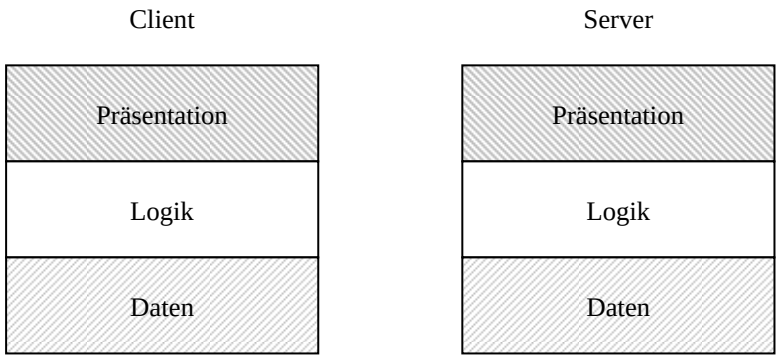


Abbildung 16 System-Architektur im Mobile Game

Abbildung 16 zeigt den Aufbau der einzelnen Schichten im Mobile Game. Diese sind sowohl auf dem Client wie auch auf dem Server gleich angeordnet, unterscheiden sich aber in der Art der Implementierung und der Funktionalität. Die Präsentationsschicht ist für die Benutzeranzeige zuständig. Diese Schicht beinhaltet neben den GUI-Klassen auch solche, die zu deren Steuerung und Organisation dienen.

Die Logikschicht beinhaltet alle Elemente, die die Steuerung der Applikation übernehmen. Dazu zählen unter anderem die *Monitor*-Klassen, welche auf bestimmte Ereignisse reagieren können.

Zuunterst im Schichtenmodel befindet sich die Datenschicht. Diese beinhaltet die eigentlichen Datenobjekte. Dazu gehören auch die Klassen, welche die Annotationen repräsentieren (*HandwritingAnnotation*, *SoundAnnotation* und *FotoAnnotation*).

4.1. Entwicklungs- und Testumgebung

Im Rahmen dieser Diplomarbeit wurden zwei verschiedene mobile Geräte verwendet, namentlich der PDA iPAQ 4700hx sowie das Smartphone Qtek 9000. Auf beiden Geräten wurde mit der J9-Runtime von IBM [IBMJ9] gearbeitet.

Je nachdem, ob man für den mobilen Client oder für den Server entwickelte, setzte man für die Entwicklung verschiedene Systeme ein. Für den Client kam das WebSphere Studio der Version 5.7.1 von IBM [IBMWebSphere] zum Einsatz. Für Entwicklungen auf der Serverseite wurde Eclipse in der Version 3.2.1 verwendet. Für die graphischen Elemente hat man sich für die OpenSource Komponente SWT [SWTToolkit] entschieden. Diese wurde bereits in der derzeitigen Implementierung vom Mobile Game verwendet.

Des Weiteren wurden für die Ansteuerung der Kamera und der Sprachaufzeichnung C++ Bibliotheken entwickelt, welche mit Visual Studio 2005 von Microsoft erstellt wurden.

Als Testumgebung dienten die Device-Emulatoren von Microsoft sowie der iPAQ PDA und das Qtek Smartphone.

4.2. Konzept

Ziel war es von Anfang an, die Annotationen so zu implementieren, dass sie einfach und effizient erweitert werden können. Abbildung 17 zeigt die Klassenhierarchie, die entwickelt wurde, um eine möglichst flexible und einfach erweiterbare Struktur zu realisieren. Die Abbildung zeigt den Aufbau der Annotationen in der Datenschicht. Als Basis dient die Klasse *AnnotationBase*. (Von dieser Basisklasse werden alle anderen Annotationsklassen abgeleitet.) Sie beinhaltet die wichtigsten und für alle Annotationen gültigen Eigenschaften. Dazu gehören Position, Titel, ID, Teamname und ein Verweis auf ein *Reply*-Objekt. Dieses Objekt wird verwendet, um eine Annotation zu annotieren. Ein *Reply* bezieht sich immer auf eine bestehende Annotation.

Abbildung 17 Klassendiagramm Annotationen - Datenschicht

Alle Klassen implementieren die Serializable Schnittstelle. Diese wird benötigt, um die Instanzen der Klassen später über das Netzwerk versenden zu können. Die Klasse *Annotation* repräsentiert die normalen Textannotationen, die bereits vor dieser Arbeit im Mobile Game implementiert waren. Sie wurden jedoch dahingehend angepasst, dass sie in die neue Struktur passen. Die Klasse *BLOBAnnotation* stellt die Basis für die Handschrift-, Voice- und Foto-Annotationen dar und verfügt über die entsprechenden

Eigenschaften, die von allen neuen Annotationen gebraucht werden. Dazu gehören die eigentlichen Daten sowie die Angabe der Datengrösse und des Datenoffsets. Die Daten werden dabei als binäre Arrays repräsentiert. Zuletzt wird noch ein Indikator (*is_transfered*) verwendet, der besagt, ob die Daten vollständig transferiert wurden oder nicht.

4.3. Entwicklungsmethode

Als Entwicklungsmethode wurde Prototyping gewählt. Es war schon zu Beginn ersichtlich, dass die Entwicklung in viele Bereiche vorstossen würde, die es unmöglich machten, genügend Vorwissen zu sammeln, um ein anderes Programmierparadigma anzuwenden. So gab es zum Beispiel Informationen darüber, wie man die Kamera ansteuern konnte. Es war aber unmöglich vorherzusagen, wie sich das im Zusammenhang mit J9 von IBM umsetzen liess. Deshalb wurde grossen Wert darauf gelegt, für alle diese kritischen Bereiche Prototypen zu entwickeln und diese dann zu testen.

4.4. Serialisierung

Das Serialisieren in Java bietet die Möglichkeit, Instanzen von Klassen in eine Folge von Bytes umzuwandeln. Diese serielle Form der Objekte wird benötigt, um diese über das Netzwerk übertragen zu können. Dazu müssen die Objekte, die serialisiert werden sollen, die Schnittstelle *Serializable* implementieren [JavaSerialize].

```
private void send()
{
   int port = 4400;
   String ipAdress = "192.168.1.1";
   Socket connection = null;
   ObjectOutputStream oos = null;
   try{
      connection = new Socket(ipAdress, port);
      oos = new ObjectOutputStream(connection.getOutputStream());
      oos.writeObject(new String("this message was sended over tcp"));
      oos.close();
      connection.close();
   } catch (Exception e){
   }
}
```

Quellcode 1 Serialisierung im Mobile Game

Quellcode 1 zeigt, wie die Serialisierung im Mobile Game eingesetzt wird. Im Beispiel wird ein *String*-Objekt erzeugt und über das Netzwerk an den Empfänger mit der IP Adresse 192.168.1.1 versendet. Anstelle des *String*-Objektes können beliebige Objekte

angegeben werden, die die Schnittstelle *Serializable* implementieren. Analog dazu funktioniert auch das Deserialisieren. Dabei wird anstelle von der *writeObject*-Methode die *readObject*-Methode verwendet.

Obwohl die Klasse *BLOBAnnotation* die Schnittstelle *Serializable* implementiert, zeigt Quellcode 2, dass die Eigenschaften *data*, *data_offset* und *is_transfered* als *transient* gekennzeichnet sind.

```
private transient int data_offset = 0;
private int data_size = 0;
private transient boolean is_transfered = false;
private transient byte[] data = null;
```

Quellcode 2 Eigenschaften der BLOBAnnotation-Klasse

Das bedeutet, dass diese Eigenschaften vom Serialisierungsprozess ausgeschlossen sind. Sie werden also nicht in eine binäre Form gebracht und können somit auch nicht über das Netzwerk übertragen werden.

4.5. Datenkomprimierung mit Java

Um die Grösse der Daten zu reduzieren wurden die von Java zur Verfügung gestellten Klassen *GZIPOutputStream* und *GZIPInputStream* verwendet. Quellcode 3 zeigt die Komprimierungsmethode, wie sie ins System implementiert wurde.

```
/**
 * Komprimiert das byte-Array data und liefert das komprimierte
 * Array zurück.
 * @param data Byte-Array welches komprimiert werden soll
 * @return Komprimiertes Array
 */
public static byte[] compress(byte[] data)
{
   byte[] zipped_data = null;
   try
   {
      ByteArrayOutputStream out = new ByteArrayOutputStream();
      GZIPOutputStream zout = new GZIPOutputStream(out);
      zout.write(data, 0, data.length);
      zout.finish();
      zipped_data = out.toByteArray();
      out.close();
      zout.close();
   } catch (Exception e)
   {
      System.out.println(e.getMessage());
   }
   return zipped_data;
}
```

Quellcode 3 Komprimierung

4.6. Client

In diesem Teil werden die vorgenommenen Anpassungen am bisherigen System auf der Clientseite beschrieben und die neuen Entwicklungen aufgezeigt, die im Rahmen dieser Arbeit durchgeführt wurden.

Das bisherige Mobile Game beinhaltete bereits normale Text-Annotationen, die durch die Klasse *Annotation* repräsentiert wurde. Der erste Schritt der Arbeit bestand darin, die in Abbildung 17 dargestellte Klassenhierarchie auf der Datenschicht zu implementieren und den Mobile Game Client dementsprechend anzupassen. Dazu musste die Klasse *Annotation* so verändert werden, dass diese neu von der Klasse *AnnotationBase* erbt. Dies wurde erforderlich, weil alle Annotationen eine gemeinsame Basisklasse haben sollten, um sie gemeinsam behandeln zu können.

Als nächster Schritt wurden die graphischen Elemente der Präsentationsschicht eingebaut. Dazu mussten für die Handschrift-, Voice- und Foto-Annotation neue Renderer-Klassen geschrieben werden. Diese sind für das Zeichnen der Icons auf der Karte zuständig. Entworfen wurden *HandwriteAnnotationPointRenderer*, *NGSoundAnnotation-PointRenderer* und *NGPhotoAnnotationPointRenderer*. In *NGMapRenderer* wurden dann die zusätzlichen Annotationen eingetragen, damit diese später auf der Spielkarte sichtbar werden.

In der Klasse *NGMapCanvas* wurden neue Menuelemente erzeugt, um die neuen Annotationen erstellen zu können.

Abbildung 18 zeigt die neu entwickelten GUI-Dialoge und deren Vererbungshierarchie.

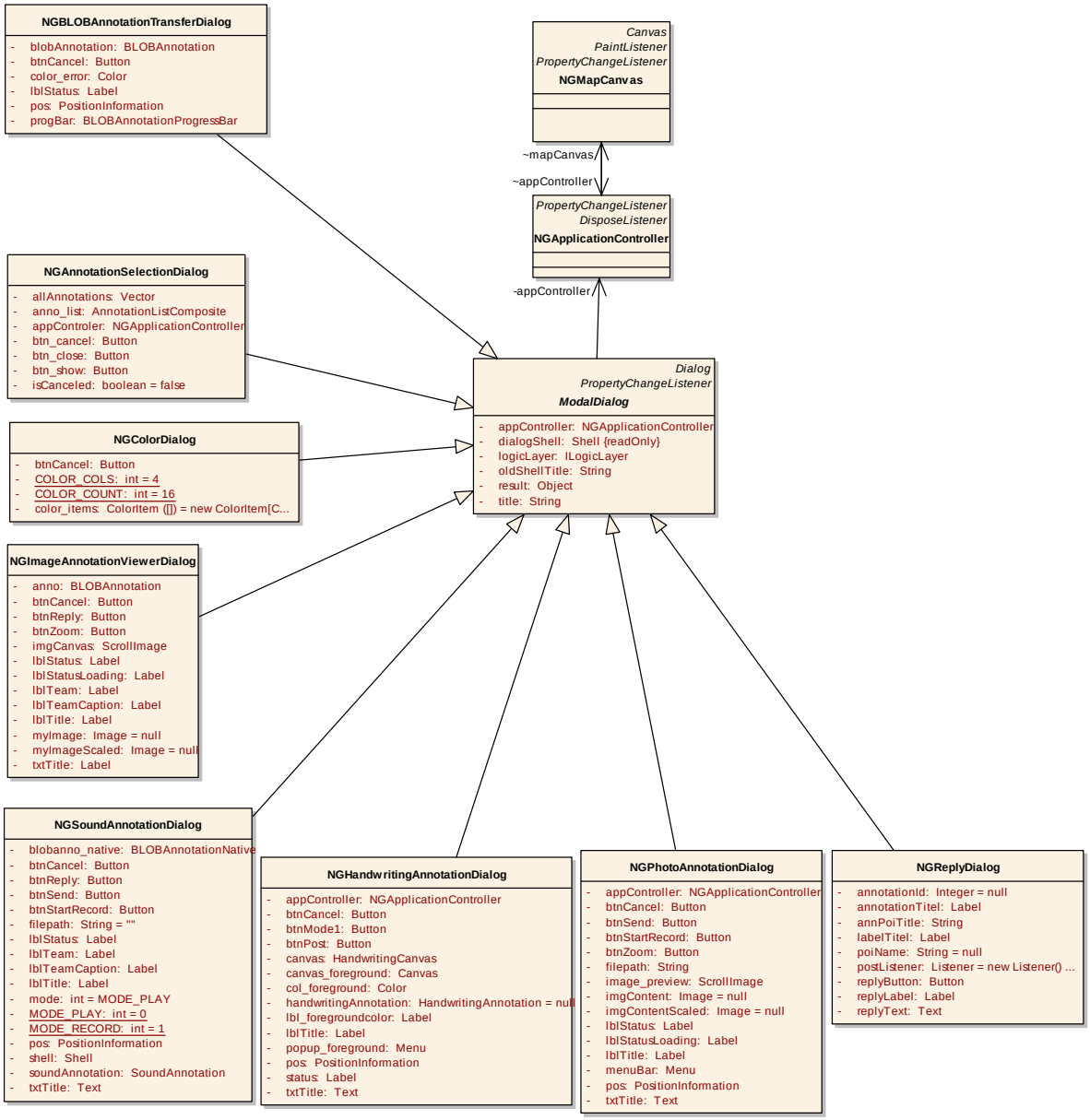


Abbildung 18 Klassendiagramm Annotationen - Präsentationschicht[7]

Zuletzt wurde die Logikschicht implementiert. Diese ist zuständig für das Versenden und Empfangen von Annotationen. Abbildung 19 zeigt die Logikschicht aus Sicht der Handschrift-, Voice- und Foto-Annotationen. Die Klasse *BLOBAnnotationLogic* implementiert die entsprechende Funktionalität dieser Annotationen.

[7] Um die Übersicht im Diagramm nicht unnötig zu erschweren, wurde für alle Klassen die Beschriftung der Operatoren weggelassen. Für die Klassen *NGApplicationController* und *NGMapCanvas* wurde zudem die Beschriftung der Attribute weggelassen.

Abbildung 19 Klassendiagramm Annotationen - Logikschicht

4.6.1. Kamera und Sprachaufzeichnung

Mit der Programmiersprache Java war es nicht möglich, hardwarenahe Funktionen des mobilen Gerätes auszuführen. Dies war aber erforderlich, da in der Aufgabenstellung dieser Arbeit vorausgesetzt wurde, Sprachaufzeichnungen aufnehmen und Fotos machen zu können. So musste auf eine andere Möglichkeit ausgewichen werden um die Funktionalität zu implementieren.

Zu diesem Zweck wurde eine C++ Bibliothek entwickelt, welche die entsprechenden Funktionen implementiert.

Funktionsname	**Beschreibung**
startRecording(String filepath)	Startet die auf dem PDA integrierte VoiceRecorder-Komponente und speichert die Aufnahme an der angegeben Position.
startPlaying(String filepath)	Startet die VoiceRecorder-Komponente und gibt die an der Position *filepath* abgelegte Datei wieder.
setLocation(int x, int y)	Legt die Position der VoiceRecorder-Komponente fest.
setHandle(int handle)	*handle* ist das Fensterhandle des übergeordneten Fensters. Die VoiceRecorder- und die Fotokamera-Komponente benötigen diesen Wert für die modale Anzeige.
startCamera(String filepath)	Startet den in Windows CE 5.0 integrierten Kamera-aufnahme-Dialog und speichert die Aufnahme an der mit *filepath* spezifizierten Position. Diese Methode ist nur in der Win CE 5 Bibliothek ausprogrammiert. In den anderen Versionen ist nur die Schnittstelle definiert.
handleSIP(int command)	Durch diese Funktion kann man die virtuelle Tastatur der PDAs steuern. Gültige Werte für *command* sind: 0 Anzeigen 1 Unterdrücken Diese Methode wir derzeit im Mobile Game nicht

	verwendet und wurde ausschliesslich zu Testzwecken implementiert.

Tabelle 2 Interface Funktionen von BLOBAnnotationNative

Tabelle 2 zeigt die Funktionen, die in der Bibliothek vorhanden sind. Java bietet dabei das Konzept von JNI[8] an, um nativen Code in der J9 Umgebung auszuführen. JNI definiert eine standardisierte Programmierschnittstelle, über welche die Funktionen der C++ Bibliothek aufgerufen werden können.

```
JNIEXPORT jint JNICALL Java_annotation_BLOBAnnotationNative_startPlaying
        (JNIEnv *env, jobject, jstring str){
     }
```

Quellcode 4 JNI Interface

Quellcode 4 zeigt einen Ausschnitt aus der entwickelten plattformabhängigen C++ DLL. Der Aufbau des Methodennamens zeigt, dass darin auch Informationen zu der aufrufenden Klasse vorhanden sein müssen. Die Klasse heisst *BLOBAnnotationNative* und befindet sich im Package *annotation*.

```
     public native int startRecording(String filepath);
     public native int startPlaying(String filepath);
     public native int setLocation(int x, int y);
     public native int startCamera(String filepath);
     public native int setHandle(int handle);
```

Quellcode 5 JNI Interface

In Quellcode 5 werden in Java die externen Funktionen definiert. Diese Definitionen befinden sich ebenfalls in der Klasse *BLOBAnnotationNative*. Da das Kompilat dieser Bibliothek plattformspezifisch ist, muss man für verschiedene Plattformen verschiedene DLL-Versionen verwenden. Das bedeutet, dass eine Entwicklung unter Windows XP nur dann möglich ist, wenn ein entsprechendes Kompilat vorliegt. Zu diesem Zweck wurde eine leere Bibliothek entwickelt, welche nur den Interfaceteil implementiert, nicht aber die Funktionalität. Die Klasse *BLOBAnnotationNative* entscheidet zur Laufzeit, welche Bibliothek geladen wird.

```
public BLOBAnnotationNative()
{
     //determine platform for loading dll
     String osname = System.getProperty("os.name");
     if (osname.equals("Windows CE"))
     {
        //Load CE native library
        String osversion = System.getProperty("os.version");
```

[8] JNI engl. Java Native Interface

```
        if (osversion.charAt(0) == '5')
        {
           System.loadLibrary("mGame_native_wince5.dll");
        } else
        {
           System.loadLibrary("mGame_native_wince4.dll");
        }
     }
     else
     {
        //load win32 library (empty methods inside dll)
        //used for testing and development
        System.loadLibrary("mGame_native_win32.dll");
     }
}
```

Quellcode 6 Laden der JNI Bibliothek

Die Klasse *BLOBAnnotationNative* kapselt diese Aufrufe und definiert die entsprechenden Funktionen auf der Java-Seite. Quellcode 6 zeigt, wie die Bibliothek, abhängig vom Betriebssystem, ins Mobile Game geladen wird. Um auch auf normalen Computern mit dem Web Sphere Studio arbeiten zu können, wurde eine Bibliothek für Windows 32 Systeme entwickelt. Diese beinhaltet alle Interfacedefinitionen wie die anderen DLLs, implementiert diese jedoch nicht und liefert immer den Wert 0 zurück.

4.6.2. Handschrift-Annotation

Die Handschrift-Annotationen sollten mit dem Eingabestift der PDAs erstellt werden können. Dazu musste ein Zeichnungsprogramm entwickelt werden, welches die Position des Stiftes (der Maus) ermittelt und dementsprechend eine Linie zeichnet. Dieses Zeichnungsprogramm erfüllt folgende Aufgaben:

- Erstellen einer Zeichnung
- Wählen einer Zeichnungsfarbe
- Löschen einer Zeichnung

Der Dialog *NGHandwritingDialog* kapselt die graphische Oberfläche und beinhaltet ein *HandwritingCanvas*, welches die eigentlichen Zeichnungsfunktionalitäten bereitstellt. Die Klasse *HandwritingCanvas* ist von *Canvas* abgeleitet, welche in SWT (Standard Widget Toolkit) als Standardklasse definiert ist.

In Tabelle 3 wird die Schnittstelle von *HandwritingCanvas* beschrieben.

Methodenname	**Beschreibung**
createImage(int width, int height)	Erstellt ein Image-Objekt mit angegebener Breite und Höhe. Dieses Image-Objekt wird intern verwendet, um die gezeichneten Linien aufzuzeichnen. An-

	hand der übergebenen Dimensionen werden auch die Grössen der Bildlaufleisten bestimmt.
setMode(int mode)	Setzt den Grafikmodus des Objektes. Gültige Werte sind 0 für den normalen Modus und 1 für den Löschmodus.
getMode()	Liefert den aktuellen Betriebsmodus zurück.
setDoDrawLine(boolean doDrawLine)	Dient als Indikator, ob die nächste Linie gezeichnet werden soll oder nicht. Diese Funktion hat vor allem dann Bedeutung, wenn während des Zeichnens ein neues Fenster den Fokus erhält. Dies tritt beispielsweise beim Auswählen der Zeichnungsfarben auf.
setColorForeground(Color col_foreground)	Setzt die Zeichnungsfarbe.
setColorDelete(Color col_delete)	Setzt die Farbe des Rechtecks, das im Löschmodus angezeigt wird.

Tabelle 3 Schnittstelle von HandwritingCanvas

Die *HandwritingCanvas* Klasse unterstützt zwei unterschiedliche Betriebsmodi für das Zeichnen. Der normale Modus wird verwendet, um einfache Linien in der ausgewählten Farbe zu zeichnen. Im Löschmodus hingegen wird der ausgewählte Bereich gelöscht. Um die Positionen der Linien zu bestimmen, werden die Mausbewegungen und Mausklicks aufgezeichnet und entsprechend ausgewertet.
Die Klasse *HandwritingAnnotation* repräsentiert eine solche Handschrift-Annotation und kapselt alle benötigten Eigenschaften.

4.6.3. Voice-Annotation

Für die Aufzeichnung der Sprachnachrichten kam die entwickelte C++ Komponente zum Einsatz. Diese stellt folgende grundlegenden Funktionen zur Verfügung:

- Aufzeichnen von Audio
- Abspielen von Audio
- Setzen des Fensterhandles

Dabei ist festzuhalten, dass diese Voice-Komponente nur das WAV[9]-Format unterstützt.

Für das Erstellen einer Voice-Annotation wurde eine GUI[10]-Klasse entworfen (*NGSoundAnnotationDialog*), die von der Klasse *ModalDialog* abgeleitet ist. Dieser Dialog kann sich in zwei verschiedenen Modi befinden. Der erste Modus ist der Abspielmodus, der zweite ist der Aufnahmemodus. Im ersten werden Titel und Teamname angezeigt und die Audiodatei abgespielt. Im zweiten Modus wird ein Eingabefeld für einen Titel und eine Schaltfläche für den Aufnahmebeginn angezeigt.

Wenn die Schaltfläche für den Start der Aufzeichnung betätigt wird, wird zuerst die Methode *setHandle* und danach *startRecording* aus der C++ Bibliothek aufgerufen. Mit *setHandle* wird das aktuelle Fensterhandle übergeben, welches in der Bibliothek verwendet wird, um die Voice-Recorder-Komponente modal anzuzeigen. Die Methode *startRecording* übergibt den Pfad zu einer Datei, in welche die Aufzeichnung gespeichert werden soll und zeigt dann die Voice-Recorder Komponente der Windows Mobile Umgebung an. Nach Abschluss der Aufzeichnung liefert *startRecording* den Rückgabewert 0, falls der Vorgang erfolgreich war und nicht abgebrochen wurde. Die Aufzeichnung ist danach in der angegebenen Datei gespeichert und wird zu einem späteren Zeitpunkt in das Bytearray der Klasse *BLOBAnnotation* geladen.

Der Modus wird beim Erstellen des Dialoges durch den Aufruf des Konstruktors festgelegt. Wenn der Konstruktor mit einer gültigen *SoundAnnotation* aufgerufen wird, wird der Abspielmodus ansonsten der Aufnahmemodus gestartet.

Eine Voice-Annotation wird in der Klasse *SoundAnnotation* repräsentiert. Diese Klasse ist von *BLOBAnnotation* abgeleitet und implementiert nur zwei Konstruktoren. Beim Versenden der Annotation wird die Aufzeichnung, wie bereits beschrieben, als Bytearray in der Basisklasse *BLOBAnnotation* gespeichert. Dies geschieht, indem eine neue *SoundAnnotation* Instanz mit den Audiodaten erzeugt wird.

4.6.4. Foto-Annotation

Für das Erstellen von Foto-Annotationen wurde ebenfalls eine neue GUI-Klasse entworfen, die jedoch derjenigen der Voice-Annotationen sehr ähnlich ist. Die Klasse

[9] Format zum Speichern von Audiodaten

[10] GUI engl. Graphical User Interface – dt. Grafische Benutzeroberfläche

NGPhotoAnnotationDialog ist von der Klasse *ModalDialog* abgeleitet. Im Gegensatz zu *NGSoundAnnotationDialog* gibt es hier keine Modi. Der Dialog besteht aus einem Eingabefeld für einen Titel sowie einer Schaltfläche für das Erstellen eines Fotos. Beim Betätigen der Schaltfläche werden die Methoden *setHandle* und *startCamera* der C++ Bibliothek aufgerufen. Mit *setHandle* wird das aktuelle Fensterhandle angegeben und *startCamera* zeigt den in Windows Mobile 5 integrierten Kameradialog an. Zusätzlich wird mit *startCamera* der Pfad zu einer Datei übergeben, in welcher das Foto gespeichert werden soll. Das Fensterhandle wird benötigt, um den Kameradialog modal anzuzeigen. Nach Abschluss des Fotografierens liefert *startCamera* einer der folgenden Werte zurück:

0: Kein Fehler. Die Aufnahme wurde erfolgreich durchgeführt

-1: Die Aufnahme wurde durch den Benutzer abgebrochen.

-2: Ein ungültiges Argument wurde übergeben.

-3 Nicht genügend Speicher vorhanden.

-4 Anderer unbekannter Fehler.

Wenn kein Fehler aufgetreten ist, wurde die Foto-Aufnahme in der angegebenen Datei gespeichert. Beim Versenden der Annotation wird ein *PhotoAnnotation* Objekt mit den Bilddaten erzeugt. Die Foto-Annotation wurde in der Klasse *PhotoAnnotation* implementiert, die von der Klasse *BLOBAnnotation* abgeleitet ist. Sie implementiert nur zwei Konstruktoren. Die eigentlichen Bilddaten werden als Bytearray in der Basisklasse *BLOBAnnotation* gehalten.

4.6.5. Übertragen von Annotationen

Die drei Dialoge zum Erzeugen von Annotationen verfügen alle über die Möglichkeit, die erstellte Annotation zu versenden. Versenden bedeutet, dass die Annotation an den Server übermittelt und anschliessend an die anderen Clients verteilt wird. In der Logikschicht ist die *GameStore*-Klasse dafür zuständig, das aktuelle Spiel zu verwalten. Dazu gehört auch das Speichern aller auf dem Client vorhandenen Annotationen.

Beim Versenden wird auf die Logikschicht zugegriffen, um die entsprechende Annotation zu erzeugen und mit zusätzlichen Informationen zu versehen. Um über das Netzwerk zu kommunizieren, bedarf es eines Satzes von eindeutigen Meldungen, damit beide Kommunikationspartner wissen, was der Inhalt der Netzwerknachricht ist. Zu

diesem Zweck wurde die Klasse *GameEvents* mit neuen Ereigniskennungen erweitert. Tabelle 4 beschreibt diese neuen Kennungen.

Ereigniskennung	**Beschreibung**
NEW_BLOB_ANNOTATION_EVENT	Wird verwendet, wenn es sich bei der Netzwerknachricht um eine neue Annotation handelt. Tritt bei der Kommunikation zwischen Client-Server und Server-Client auf.
NEW_BLOB_ANNOTATION_ID_EVENT	Die Netzwerknachricht beinhaltet eine eindeutige Kennnummer. Tritt nur zwischen Server-Client auf.
NEW_BLOB_ANNOTATION_DATA_ EVENT	Wird verwendet, um Datenblöcke zwischen Client-Server und Server-Client auszutauschen.
NEW_BLOB_ANNOTATION_DATA_ ACK_EVENT	Bestätigungsmeldung in der Server-Client-Kommunikation.
NEW_BLOB_ANNOTATION_DATA_ REQUEST_EVENT	Wird verwendet, wenn der Client Daten vom Server herunterladen will.
NEW_BLOB_ANNOTATION_REPLY_ EVENT	Tritt dann auf, wenn ein *Reply*-Objekt für eine Annotation erstellt wurde. Bestandteil der Server-Client- und Client-Server-Kommunikation.

Tabelle 4 Netzwerkereignisse bei Annotationen

Die Klasse *BLOBAnnotationLogic* ist für die Annotationen zuständig und verfügt über die Methoden *createHandwritingAnnotation*, *createSoundAnnotation* und *createPhoto-Annotation*. Jede Methode der entsprechenden Annotation instanziiert entweder ein *HandwritingAnnotation*, ein *SoundAnnotation* oder ein *PhotoAnnotation*-Objekt. Diesem Objekt werden die Positionsinformationen und der Teamname des Erstellers hinzugefügt. Danach wird die *BLOBAnnotation* in die Warteschlange des GameStores mit der Methode *setWaitingBLOBAnnotation* eingefügt. Gleichzeitig wird dem Server die *BLOBAnnotation* übermittelt (ohne binäre Daten – siehe Kapitel 4.4) und es wird ihm mitgeteilt, dass eine neue Annotation verfügbar ist und dass diese eine eindeutige

Kennnummer benötigt. Die Annotation verweilt solange in der Liste, bis der Server eine entsprechende Kennnummer zur Verfügung gestellt hat. In Kapitel 3.3.3 wird der Vorgang genauer beschrieben. Die ganze Kennnummeranfrage wird asynchron ausgeführt. Das bedeutet, dass die *create*-Methode der *BLOBAnnotationLogic*-Klasse beendet ist, bevor die Annotation eine entsprechende Kennnummer erhalten hat. Der Rückgabewert der Methode ist eine Instanz der entsprechenden Annotation.

Danach wird der Transfer-Dialog (*NGBLOBAnnotationTransferDialog*) angezeigt. Dieser zeigt den aktuellen Status der Übermittlung an. Beim Anzeigen des Dialoges wird die Methode *transferData* ausgeführt. In dieser Methode wird periodisch überprüft, ob die Annotation bereits über eine Nummer verfügt oder nicht. Solange die Annotation noch über keine Kennnummer verfügt, wird für den Benutzer die Meldung „waiting for id“ angezeigt. Dies wird solange wiederholt, bis eine vordefinierte Zeitspanne abgelaufen ist oder bis eine Kennnummer vergeben wurde. Falls der Vorgang erfolglos bleibt, wird die Übermittlung abgebrochen. Wenn der Server auf die Kennnummeranfrage antwortet, wird diese der Annotation zugewiesen und die Annotation wird aus der Warteschlange des *GameStores* entfernt. In der *transferData* Methode wird dann als nächstes versucht, die eigentlichen Daten der Annotation zu übertragen. Abbildung 20 zeigt die Klasse *BLOBAnnotationDataItem*, welche eine Dateneinheit repräsentiert.

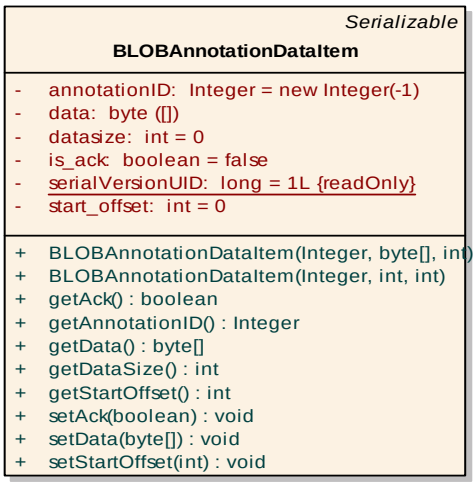


Abbildung 20 BLOBAnnotationDataItem

Im Gegensatz zu *BLOBAnnotation* sind hier die Daten nicht als *transient* gekennzeichnet und werden deshalb im Serialisierungsprozess nicht ausgeschlossen. Um die Daten der Annotation zu übertragen, wird die Methode *sendNextDataBlock* der Logikschicht aufgerufen. Dabei ermittelt *sendNextDataBlock* die Position der bereits übermittelten

Daten und versucht, den nächsten Block an den Server zu versenden. Dazu wird eine Instanz der Klasse *BLOBAnnotationDataItem* erstellt. Der Instanz wird die Kennnummer der Annotation, die Startadresse des Datenbereiches sowie der zu versendende Annotationsinhalt zugewiesen. Dieses Objekt wird dann über das Netzwerk an den Server mit der Kennung *NEW_BLOB_ANNOTATION_DATA_ITEM* versendet.
Jede Dateneinheit muss durch den Server bestätigt werden und wird deshalb, wie auch schon bei der eindeutigen Kennnummer, in eine Warteschlange gestellt. Dort verweilt der Datenblock solange, bis der Server das Element bestätigt hat oder bis eine vordefinierte Zeitspanne abgelaufen ist. Dabei liefert die Methode den Rückgabewert *true*, falls die Daten erfolgreich übermittelt wurden, beziehungsweise *false*, wenn die Übertragung fehlgeschlagen ist. Die Methode *transferData* ruft die Methode *sendNextDataBlock* solange auf, bis die gesamte Datenmenge übermittelt wurde. Zwischen den einzelnen Aufrufen wird die Anzeige aktualisiert und durch eine Fortschrittsanzeige wird der aktuelle Stand des Vorganges angezeigt.

4.6.6. Empfangen und Anzeigen von Annotationen

Wenn auf dem Server eine Annotation vollständig übermittelt wurde, wird diese an alle Clients weitergeleitet. Diese wird dann im *GameStore* auf der entsprechenden Karte gespeichert und erscheint in der Hauptansicht des Mobile Games mit einem entsprechenden Icon. Zu diesem Zeitpunkt sind die Annotationen noch leere Hüllen. Das bedeutet, dass das Datenfeld der Klasse *BLOBAnnotation* noch leer ist. Dieses Feld wurde als *transient* gekennzeichnet und somit nicht serialisiert. Durch einen Klick auf das Icon wird der *BLOBAnnotationSelectionDialog* angezeigt. Je nachdem, ob sich mehrere Annotationen in Reichweite des Klicks befinden, werden dort mehrere Einträge aufgelistet. Dieser Dialog verfügt über eine Tabelle (*AnnotationListComposite*), in welcher die entsprechenden Annotationen angezeigt werden. Jede Zeile der Tabelle kann vier verschiedene Zustände haben:

0: Annotation vollständig übermittelt.
1: Fehler.
2: Wird gerade übertragen.
3: Annotation noch nicht übermittelt.

Die Zustände werden in der Tabelle durch verschiedene Symbole gekennzeichnet. Wenn ein Element den Zustand 3 oder 1 besitzt, hat der Benutzer die Möglichkeit, diese

Annotation auf das mobile Gerät zu übertragen. Dazu erscheint bei der Auswahl der entsprechenden Zeile die Schaltfläche „Download“. Falls ein Element den Status 0 hat, wechselt die Schaltfläche von „Download“ nach „Show“. Die Annotation kann in diesem Fall angezeigt oder abgespielt werden. Der Status 2 bedeutet, dass die Übertragung gerade im Gange ist. Die Schaltfläche ist dementsprechend deaktiviert.

Download

Ist die Annotation noch nicht vollständig auf dem mobilen Gerät vorhanden, kann sie mittels eines Downloadvorganges übertragen werden. Dazu wird im Dialog *BLOBAnnotationSelectionDialog* die Methode *performDownload* ausgeführt. Diese Methode erhält als Parameter eine *BLOBAnnotation*. Auf der Logikschicht wird dann für diese Annotation die Methode *requestNextDataBlock* solange aufgerufen, bis die Übertragung vollständig durchgeführt wurde. Diese Methode sendet eine Anfrage an den Server, welche die Startadresse und die Datenmenge der gewünschten Annotation beinhaltet. In Kapitel 3.3.5 wird der Übermittlungsvorgang ausführlich beschrieben.
Bei einem erfolgreichen Download wechselt der Status des Eintrages auf 0. Wenn der Vorgang nicht erfolgreich war, wechselt der Status auf 1.

Anzeige

Ist die Annotation vollständig auf dem mobilen Gerät vorhanden, kann sie, je nach Annotationstyp, angezeigt oder abgespielt werden. Handelt es sich um eine Handschrift- oder Foto-Annotation wird der Dialog *NGImageAnnotationViewerDialog* angezeigt. Dieser lädt die Bilddaten in ein *Image*-Objekt und erzeugt gleichzeitig ein skaliertes Vorschaubild. Dieses wird mittels einer *ScrollImage* Komponente angezeigt. Diese Komponente besitzt zwei Bildlaufleisten und ermöglicht somit das Bild zu scrollen, falls die Dimensionen des Bildes grösser als diejenigen der Anzeigefläche sind.
Bei einer Voice-Annotation wird der *NGSoundAnnotationDialog* angezeigt. Dieser wechselt, wie in Kapitel 4.6.3 beschrieben, in den Abspielmodus und spielt die Audiodaten ab.

4.7. Server

Wie beim Client ist auch die Systemarchitektur auf dem Server in drei Schichten aufgeteilt (Präsentations-, Logik- und Datenschicht). Der Aufbau der Datenschicht ist sowohl beim Server als auch beim Client identisch.

4.7.1. Empfangen, Weiterleiten und Versenden von Annotationen

Auf dem Server musste in einem ersten Schritt die Logikschicht für die neuen Annotationen angepasst werden. Zu diesem Zweck wurde die Klasse *BLOBAnnotationHandler* entworfen, welche in Abbildung 21 dargestellt ist. Die Klasse implementiert zudem die *IMessageListener* Schnittstelle, um auf eingehende Netzwerknachrichten zu reagieren.

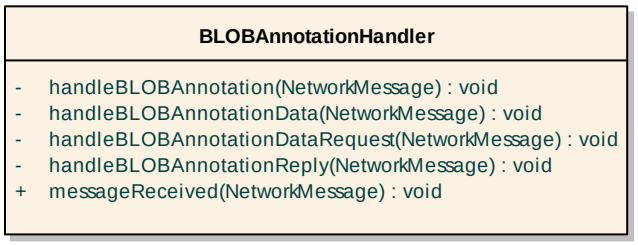


Abbildung 21 Server - Logikschicht

BLOBAnnotationHandler stellt vier Methoden zur Verfügung, um die entsprechenden Netzwerkereignisse zu behandeln.

Empfang von Annotationen

Wenn der Server von einem Client eine neue Annotation empfängt (wenn er die Netzwerkmeldung *NEW_BLOB_ANNOTATION_ EVENT* aus Tabelle 4 empfängt), wird die Methode *handleBLOBAnnotation* aufgerufen. Die übergebene *NetworkMessage* enthält dann bereits eine Instanz einer Annotation, welche auf dem Client erstellt wurde. Zu diesem Zeitpunkt verfügt die Annotation aber noch über keine eindeutige Kennnummer. In einem ersten Schritt wird diese Kennnummer durch den Server vergeben und an den Client zurückgesendet. Dies geschieht, indem der Server ein neues *NetworkMessage*-Objekt erzeugt, diesem die Nummer sowie das Ereignis *NEW_BLOB_ANNOTATION-_ID_EVENT* zuweist und die Netzwerkmeldung über das Netzwerk versendet. Gleichzeitig wird die empfangene Annotation in einer Liste gespeichert.

Sobald der Client die eindeutige Kennnummer erhalten hat, beginnt dieser mit der Übertragung des Annotationsinhaltes. Auf der Serverseite wird beim Eintreffen eines Datenpaketes das Ereignis *NEW_BLOB_ANNOTATION_DATA_EVENT* ausgelöst und die Methode *handleBLOBAnnotationData* aufgerufen. Die Daten des übertragenen *BLOBAnnotationDataItem*-Objekts werden der entsprechenden Annotation auf dem Server zugewiesen. Sobald die Daten abgearbeitet wurden, sendet der Server eine Bestätigungsmeldung an den Client zurück. Diese Meldung beinhaltet die Ereigniskennung *NEW_BLOB_ANNOTATION_DATA_ACK_EVENT*.
Dieser Vorgang wird so oft wiederholt, bis die Annotation vollständig übertragen wurde oder bis der Client keine weiteren Datenblöcke sendet.

Weiterleiten von Annotationen

Falls die Annotation vollständig übermittelt wurde, sendet der Server an alle registrierten Clients automatisch die Meldung *NEW_BLOB_ANNOTATION_EVENT* inklusive der zugehörigen Instanz der Annotation.

Versenden von Annotationen

Sobald ein Client die Daten einer Annotation vom Server erhalten will, wird er eine Anfragenachricht versenden. Diese Nachricht hat die Kennung *NEW_BLOB_ANNOTATION_DATA_REQUEST_EVENT*. Auf dem Server wird beim Eingang dieser Nachricht die Methode *handleBLOBAnnotationDataRequest* aufgerufen. Die vom Client übermittelte Nachricht beinhaltet ein *BLOBAnnotationDataItem*-Objekt, in welchem die Startadresse der Daten, die Datenmenge sowie die gewünschte Annotationsnummer angegeben sind. Der Server ermittelt die entsprechende Annotation und kopiert die angeforderten Daten in ein Antwort-Objekt, welches wiederum eine Instanz der Klasse *BLOBAnnotationDataItem* ist. Dieses Datenpaket wird dann mit der Kennung *NEW_BLOB_ANNOTATION_DATA* zurück an den Client versendet. Dieser wiederholt die Datenanfrage so lange, bis die Annotation vollständig übertragen wurde.

4.7.2. Erstellen und Anzeigen von Annotationen

Abbildung 22 zeigt die Präsentationsschicht auf dem Server im Zusammenhang mit den Annotationen.

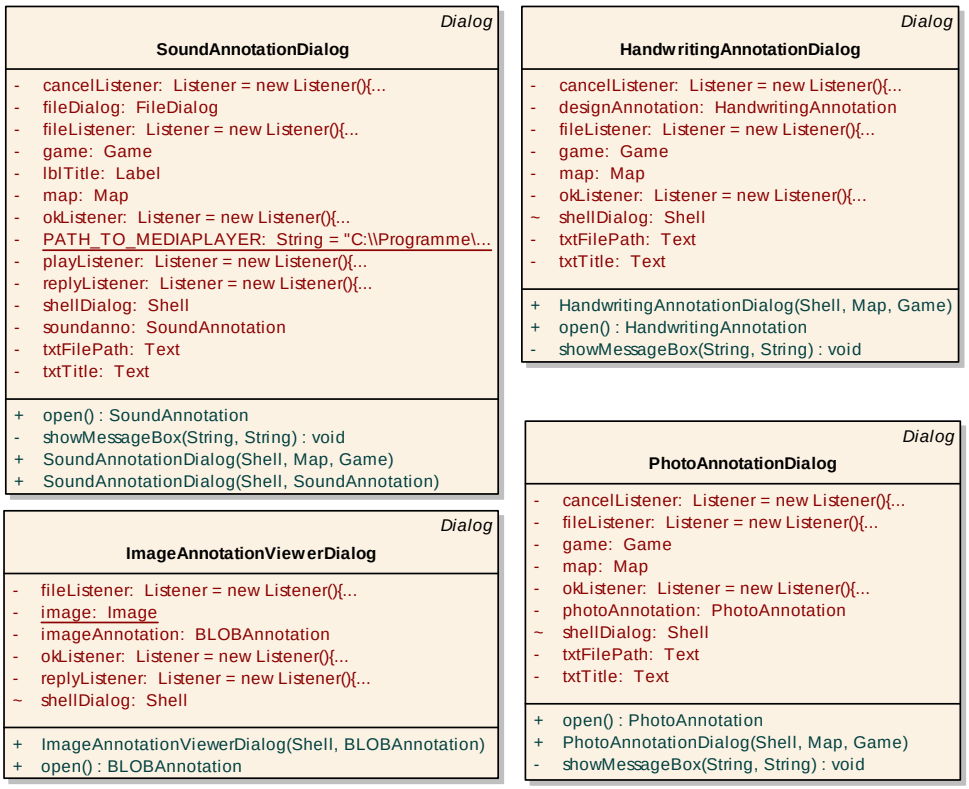


Abbildung 22 Server - Präsentationsschicht

Mit Hilfe der Dialoge *SoundAnnotationDialog*, *HandwritingAnnotationDialog* und *PhotoAnnotationDialog* können auf dem Server die entsprechenden Annotationen erzeugt werden. Dabei können Handschrift- und Foto-Annotation aus JPEG[11]-Dateien generiert werden. Für Voice-Annotationen werden WAV-Dateien verwendet.

Der Dialog *ImageAnnotationViewerDialog* dient dazu, Handschrift- und Foto-Annotationen anzuzeigen.

4.7.3. Speichern und Laden von Spielständen

Die Serverkomponente des Mobile Games ermöglicht das Speichern und Laden von Spielständen. Die zuständigen Klassen *WriteGameData* und *ReadGameData* übernehmen die entsprechenden Funktionen. Dabei wird ein *Game*-Objekt serialisiert und in eine Datei geschrieben oder aus einer Datei geladen. Da die *BLOBAnnotation*-Objekte die Daten als *transient* definiert haben, werden nur die leeren Hüllen gespeichert. Der eigentliche Annotationsinhalt wird dabei beim Serialisierungsvorgang ignoriert.

[11] Bei JPEG handelt es sich um ein Dateiformat um Bilddaten zu speichern.

Deshalb musste der Speicher- und Ladevorgang entsprechend angepasst werden. Dazu wurden zwei weitere Klassen implementiert, welche die Daten der Annotationen in einer separaten Datei speichern.

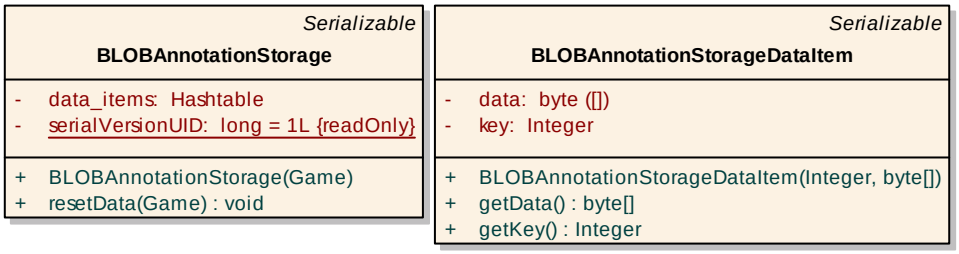


Abbildung 23 Speichern und Laden von Spielständen

Abbildung 23 zeigt die entworfenen Klassen *BLOBAnnotationStorage* und *BLOBAnnotationStorageItem*. Letztere beinhaltet die Daten einer Annotation sowie deren eindeutige Kennnummer. Es ist ersichtlich, dass das *data*-Attribut nicht als *transient* gekennzeichnet ist und somit serialisiert werden kann. *BLOBAnnotation-Storage* führt eine Tabelle aller Dateneinheiten und ist selbst auch wieder serialisierbar. Beim Speichern und Laden wird neben dem *Game*-Objekt auch das *BLOBAnnotationStorage*-Objekt in eine Datei gespeichert beziehungsweise aus einer Datei geladen.

4.7.4. Exportieren von Annotationen

Um die Annotationen auch ausserhalb des Mobile Games verwenden zu können, wurde eine Exportfunktion implementiert. Die Klasse *AnnotationExport* erstellt dazu für jede im Mobile Game vorhandene Annotation eine entsprechende Datei mit dem Annotationsinhalt. Für Handschrift- und Foto-Annotation generiert der Export JPEG-Dateien, für die Voice-Annotation werden WAV-Dateien erzeugt.

5. Anwendung

Dieses Kapitel beschreibt die grundsätzlichen Funktionen aus Sicht eines Mobile Game Benutzers im Zusammenhang mit den Annotationen.

5.1. Client

Das Kapitel über die Anwendung des Clients behandelt die grundlegenden Vorgänge für das Erstellen und Anzeigen von Annotationen.

5.1.1. Hauptansicht

Die Hauptansicht dient im Mobile Game der Orientierung auf dem Spielgebiet. Sie besteht aus einer Spielkarte, der aktuellen Position und einem Auswahlmenü für verschiedene Aktionen.

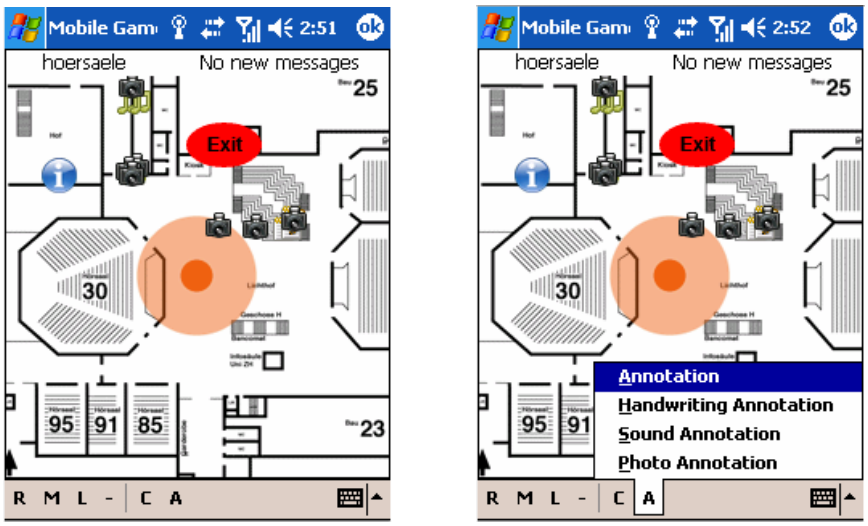


Abbildung 24 Client - Hauptansicht

Abbildung 24 zeigt die Hauptansicht im Mobile Game. Um eine Annotation zu erzeugen muss man in der Menüleiste den Menüpunkt „A“ auswählen. Im darauf folgenden Untermenü kann man die gewünschte Annotation auswählen.

5.1.2. Erstellen einer Handschrift-Annotation

Um eine Handschrift-Annotation zu erstellen, wählt man im Annotationsmenü der Hauptansicht „Handwriting Annotation“ aus.

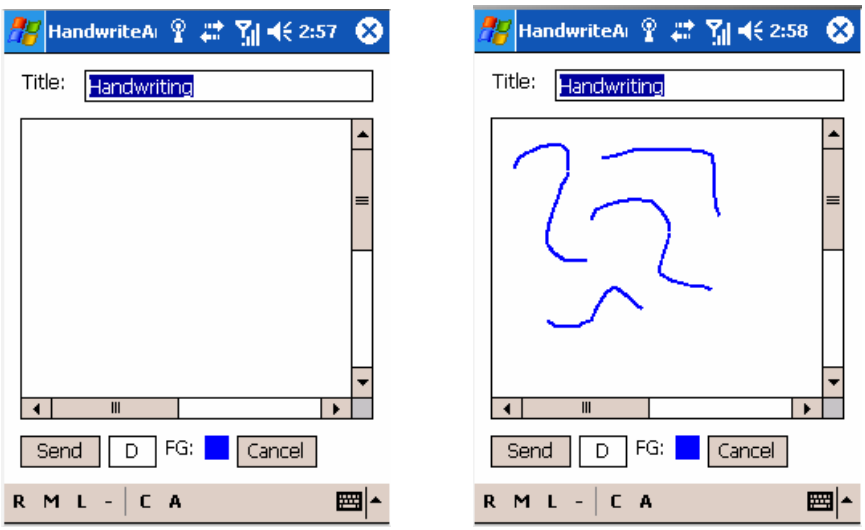


Abbildung 25 Client – Handschrift-Annotation

Abbildung 25 zeigt den Dialog für die Handschrift-Annotation. Hier kann der Titel eingegeben und in den Zeichnungsbereich gezeichnet werden. Zeichnungen erstellt man, indem der Eingabestift mit leichtem Druck über die Zeichenfläche gezogen wird.

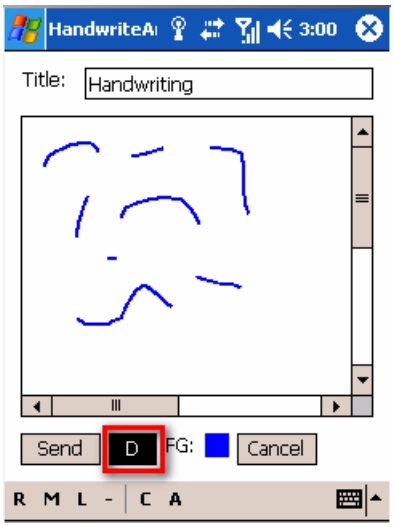


Abbildung 26 Client – Handschrift-Annotation löschen

In Abbildung 26 sieht man, wie eine Zeichnung wieder gelöscht werden kann. Dazu muss man in den Löschmodus wechseln. Dies erreicht man, indem die markierte Schaltfläche mit der Aufschrift „D“ aktiviert wird. Wenn nun mit dem Eingabestift über das Zeichnungsfeld gefahren wird, wird der Bereich unter dem Stift mit der Hintergrundfarbe übermalt. Durch einen Klick auf die Vordergrundfarbe kann man die aktuelle Zeichnungsfarbe ändern.

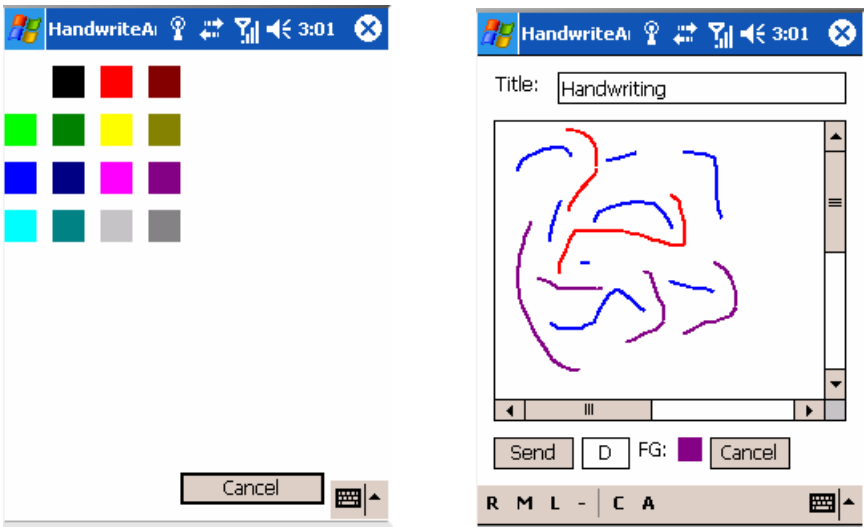


Abbildung 27 Client – Handschrift-Annotation – Farben auswählen

Abbildung 27 zeigt den Farbauswahldialog. Nach dem Ändern einer Farbe ist diese als Vordergrundfarbe ersichtlich. Die nun gezeichneten Linien werden in der entsprechenden Farbe dargestellt.

5.1.3. Erstellen einer Voice-Annotation

Um eine Voice-Annotation zu erstellen, wählt man im Annotationsmenü der Hauptansicht „Sound Annotation“ aus.

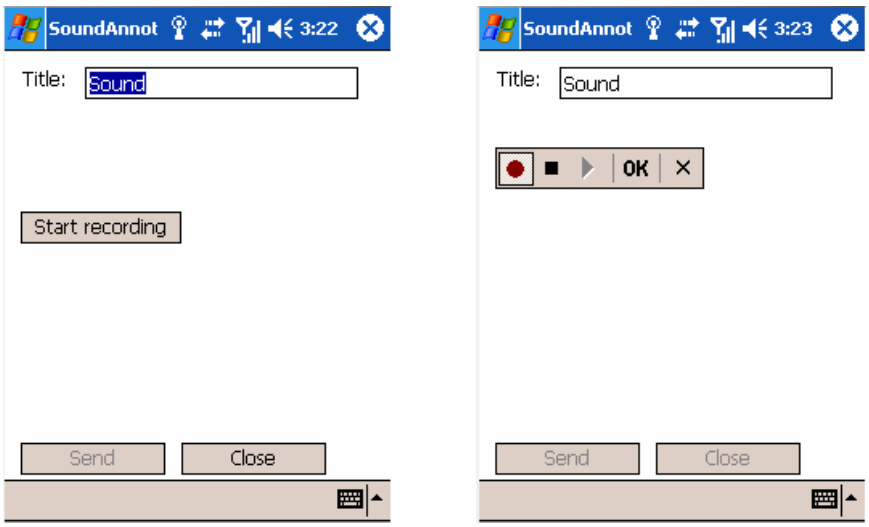


Abbildung 28 Client – Voice-Annotation

Abbildung 28 zeigt den Voice-Annotation-Dialog. Durch Betätigen der „Start recording“-Schaltfläche startet man die Aufnahme. Es erscheint dazu das in den Windows Mobile Betriebssystemen integrierte Audioaufnahmesteuerelement. Nach Abschluss der Aufnahme muss der Vorgang mit „OK“ bestätigt werden.

5.1.4. Erstellen einer Foto-Annotation

Um eine Foto-Annotation zu erstellen, wählt man im Annotationsmenü der Hauptansicht „Photo Annotation“ aus. Foto-Annotationen können nur auf Geräten ausgeführt werden, die auch über eine integrierte Digitalkamera verfügen.

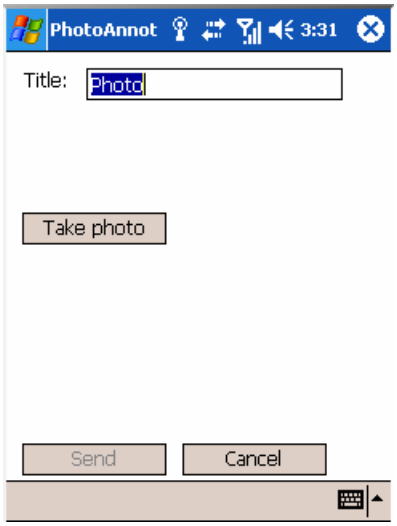


Abbildung 29 Client – Foto-Annotation

Abbildung 29 zeigt den Foto-Annotation-Dialog. Um ein Foto zu erstellen, muss die „Take photo“-Schaltfläche betätigt werden. Danach erscheint der Dialog des Betriebssystems zum Erstellen von Fotos. Nach Beendigung der Aufnahme wird man zurück in den Foto-Annotation-Dialog verwiesen.

5.1.5. Versenden von Annotationen

Jede Annotation kann nach ihrer Erstellung versendet werden. Das bedeutet, dass sie an den Server übermittelt und für andere Benutzer auf der Karte ersichtlich wird. Sobald die „Send“-Schaltfläche eines Annotationsdialoges gewählt wird, versucht der Client die Annotation an den Server zu senden. Abbildung 30 zeigt dies an einem Beispiel für Handschrift-Annotationen. Nach dem Auslösen des Vorganges wird ein Transferdialog angezeigt. Dieser liefert Informationen zum aktuellen Status der Übermittlung. Zuerst wird auf eine eindeutige Kennnummer des Servers gewartet, bevor die eigentliche Datenübertragung beginnen kann.

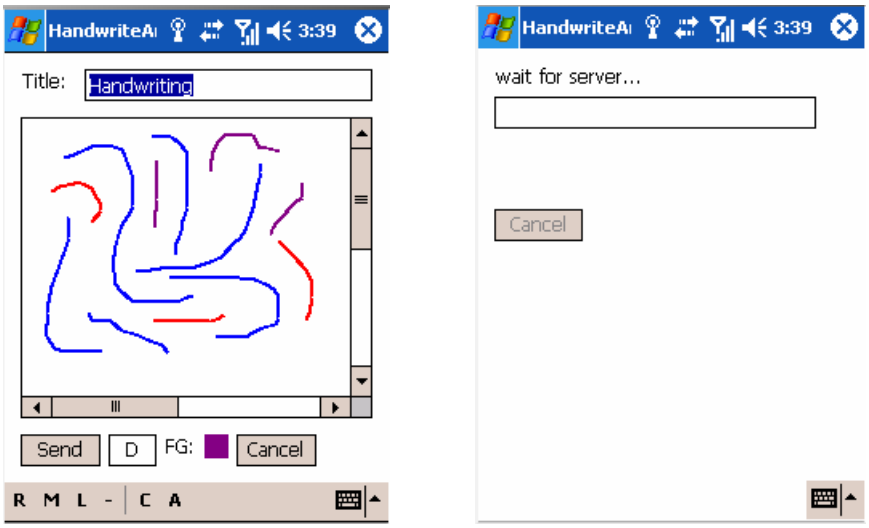


Abbildung 30 Client – Versenden von Annotationen

Falls die Anfrage für die Kennnummer nicht erfolgreich verläuft, wird eine entsprechende Meldung ausgegeben. Abbildung 31 zeigt die Ausgabe des Dialoges, wenn ein solcher Fehler aufgetreten ist. Der Übermittlungsvorgang muss abgebrochen werden, um einen neuen Versuch zu starten. Beim Abbruch gelangt man in den vorhergehenden Dialog zurück. In diesem Beispiel wäre dies der Dialog zum Erstellen von Handschrift-Annotationen.

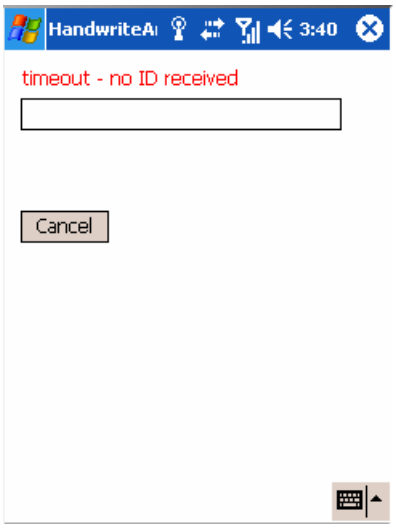


Abbildung 31 Client – Fehler bei Kennnummeranfrage

Falls die Anfrage nach der eindeutigen Kennnummer erfolgreich verläuft, beginnt die eigentliche Datenübertragung. Der Status im Transferdialog wechselt und die Beschriftung ändert sich in „transfer in progress…“. Eine Fortschrittsanzeige zeigt an, wie viel der Annotation bereits an den Server übermittelt wurde. Sollte es auch hier zu einem

Übertragungsfehler kommen, besteht wiederum die Möglichkeit, den Übermittlungsvorgang abzubrechen und erneut zu starten.

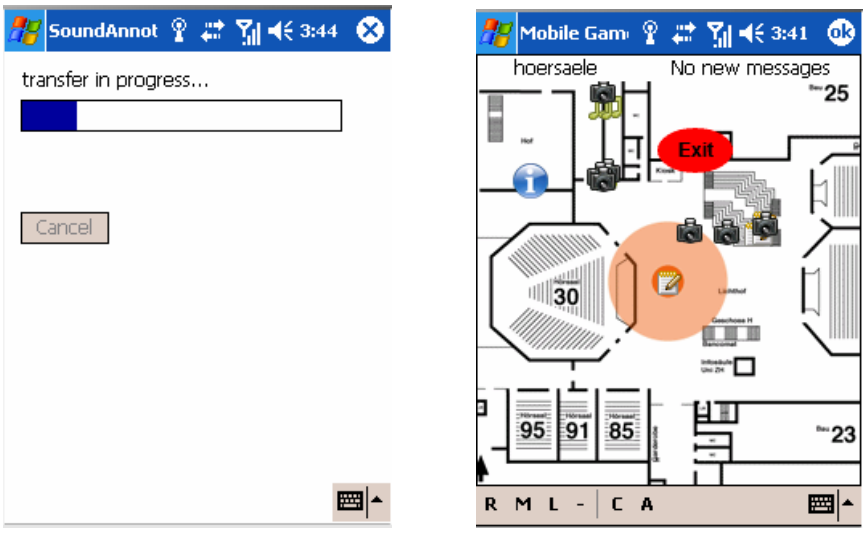


Abbildung 32 Client – Übermittlung einer Annotation

Wenn die Übertragung erfolgreich abgeschlossen werden konnte, wechselt die Anwendung automatisch zurück zur Hauptansicht. An der aktuellen Position wird dann ein Symbol der erstellten Annotation angezeigt. Andere Spieler haben nun ebenfalls die Möglichkeit, die Annotation zu betrachten.

5.1.6. Anzeigen von Annotationen

Um eine Annotation anzuzeigen, muss in der Hauptansicht auf das entsprechende Symbol der anzuzeigenden Annotation geklickt werden. Falls sich mehrere Annotationen an der gleichen Position befinden, können diese im Auswahldialog selektiert werden. Abbildung 33 zeigt den Auswahldialog mit vier verschiedenen Annotationen. Aufgrund des Symbols ist ersichtlich, dass es sich bei drei Einträgen um Foto-Annotationen handelt. Die vierte ist eine Voice-Annotation. Die mit „%“ gekennzeichnete Spalte gibt Auskunft darüber, zu wie viel Prozent der Annotationsinhalt auf dem Client verfügbar ist. Ein Wert von Null besagt, dass noch nichts vorhanden ist. Um die Daten zu übertragen, muss die „Download“-Schaltfläche betätigt werden. Die mit „S“ gekennzeichnete Spalte gibt Auskunft über den aktuellen Status des Eintrages. Wird nun der Download gestartet, wechselt das Symbol und die Übertragung der Daten beginnt. Während des Vorganges wir die Prozentanzeige aktualisiert.

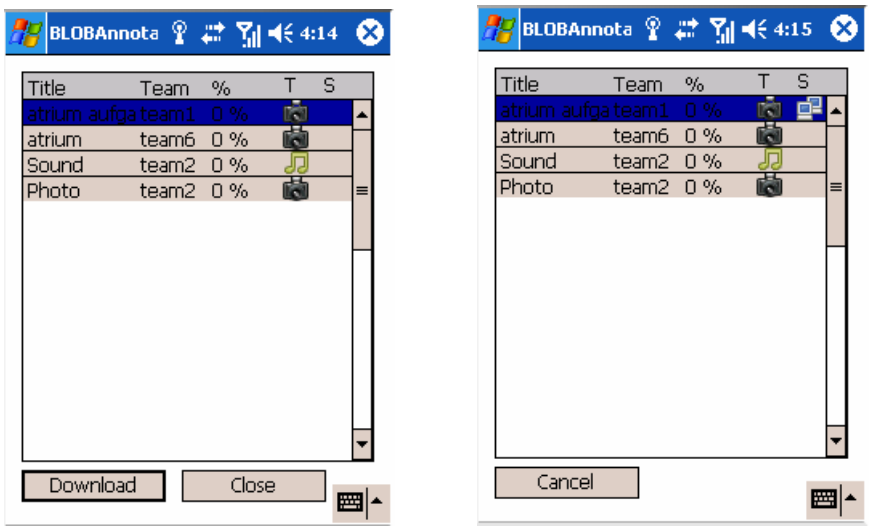


Abbildung 33 Client – Auswahldialog

Wenn die Übertragung erfolgreich abgeschlossen wurde, weist die Prozentanzeige einen Wert von 100 auf. Das Symbol in der Statusspalte wechselt auch dementsprechend. Abbildung 34 zeigt auf der linken Seite den erfolgreichen Download des ersten Eintrages an. Die Beschriftung der „Download“-Schaltfläche hat gewechselt und zeigt (je nach Annotation) „Show“ oder „Play“ an. Auf der rechten Seite der Abbildung sieht man beim zweiten Eintrag, dass die Übermittlung der Daten fehlgeschlagen ist. Dies wird mit einem entsprechenden Symbol gekennzeichnet. Die Übertragung kann jederzeit neu gestartet werden. Eine Annotation kann erst angezeigt werden, wenn sie zu 100 Prozent übertragen wurde.

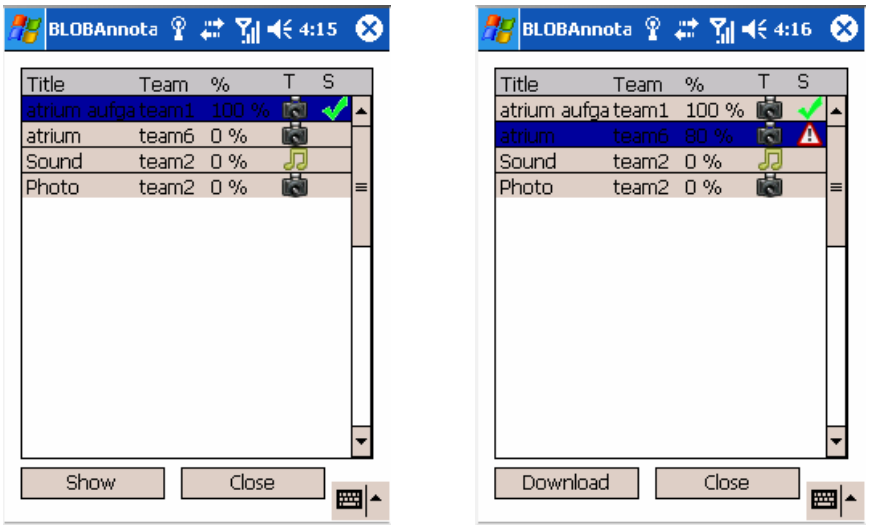


Abbildung 34 Client – Auswahldialog mit fehlerhafter Übertragung

Durch Anklicken der „Show“- oder „Play“-Schaltfläche kann man die Annotation anzeigen oder abspielen. Abbildung 35 zeigt dies am Beispiel einer Foto-Annotation. Der

Anzeigedialog zeigt neben dem Titel auch das Team an, welches die Annotation erstellt hat.

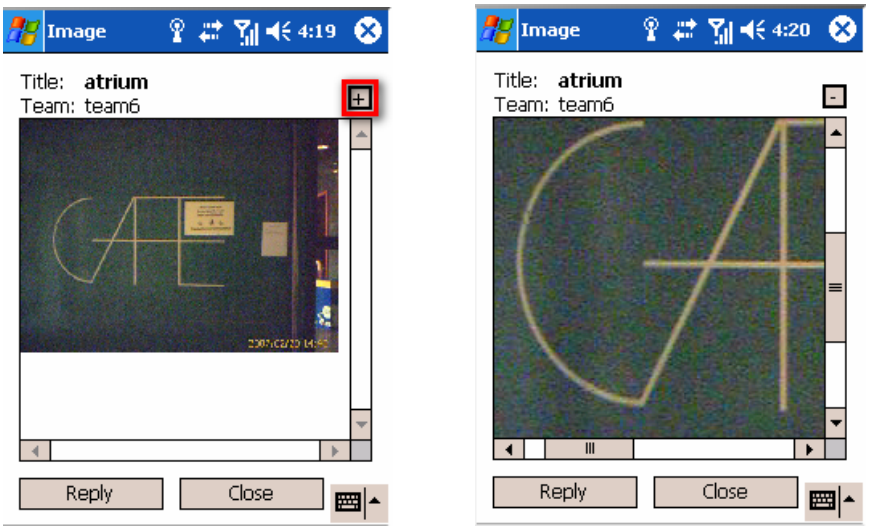


Abbildung 35 Client – Anzeige von Bild-Annotationen

Das Bild wird beim Anzeigen skaliert und als Vorschaubild dargestellt. Um das vollständige Bild anzuzeigen, muss auf die „+“-Schaltfläche geklickt werden. Im Vollbildmodus hat man die Möglichkeit, mit Hilfe der Bildlaufleisten zu scrollen. Das Anzeigen von Handschrift-Annotationen funktioniert analog.

Abbildung 36 Client – Anzeige von Voice-Annotationen

Abbildung 36 zeigt den Dialog zum Abspielen einer Voice-Annotation. Hier sind ebenfalls der Titel der Annotation und der Name des Erstellerteams ersichtlich. Mit Hilfe der integrierten Audiokomponente kann die Voice-Annotation abgespielt werden.

5.1.7. Annotieren von Annotationen

Jede Annotation kann selbst wieder annotiert werden. Jeder Anzeigedialog für Annotationen besitzt eine „Reply“-Schaltfläche. Durch Betätigen dieser Schaltfläche wechselt man in den Reply-Dialog. Abbildung 37 zeigt auf der linken Seite alle bereits erfassten Annotationen. Wenn erneut auf die „Reply“-Schaltfläche geklickt wird, wechselt man in den Reply-Erfassen-Dialog. Hier kann man eine eigene Nachricht erfassen und versenden.

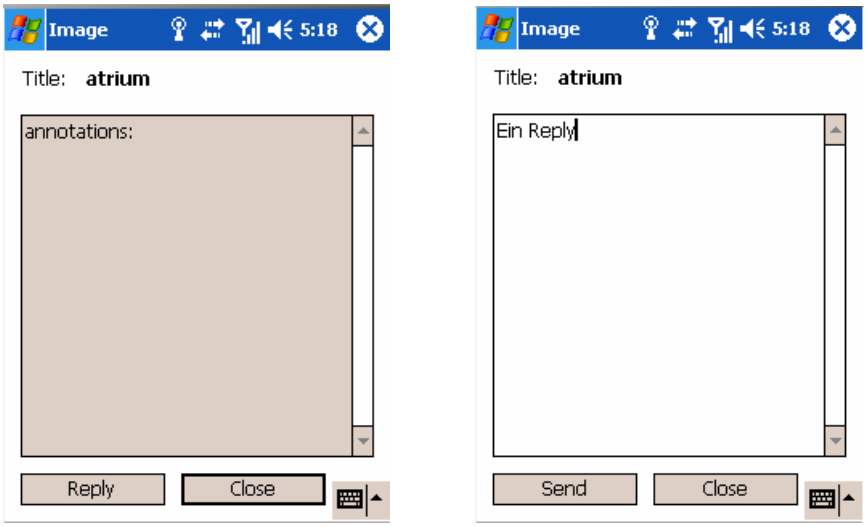


Abbildung 37 Client – Reply erfassen

Nach dem Versenden der Nachricht wechselt man zurück in den Anzeigedialog. Abbildung 38 zeigt danach die neu erfasste Reply-Nachricht an.

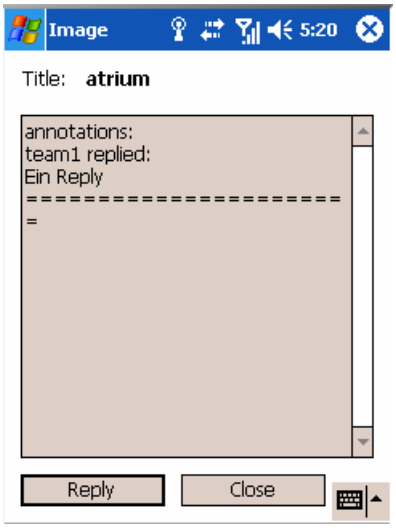


Abbildung 38 Client – Reply anzeigen

5.2. Server

Dieses Kapitel behandelt die relevanten Anwendungen auf der Serverseite. Es wird beschrieben, wie Handschrift-, Voice- und Foto-Annotationen auf dem Server erstellt und angezeigt werden können.

5.2.1. Hauptansicht

Abbildung 39 zeigt die Hauptansicht der Serverapplikation. Auf der Spielkarte sind die bereits erfassten Annotationen durch die entsprechenden Symbole dargestellt. In der Lasche „Annotations“ sind alle im Spiel vorhandenen Annotationen ersichtlich. Angezeigt wird neben der ID auch Titel, Team, Typ und ein Wert, der besagt, zu wie viel Prozent der Inhalt der Annotation auf den Server übermittelt wurde.

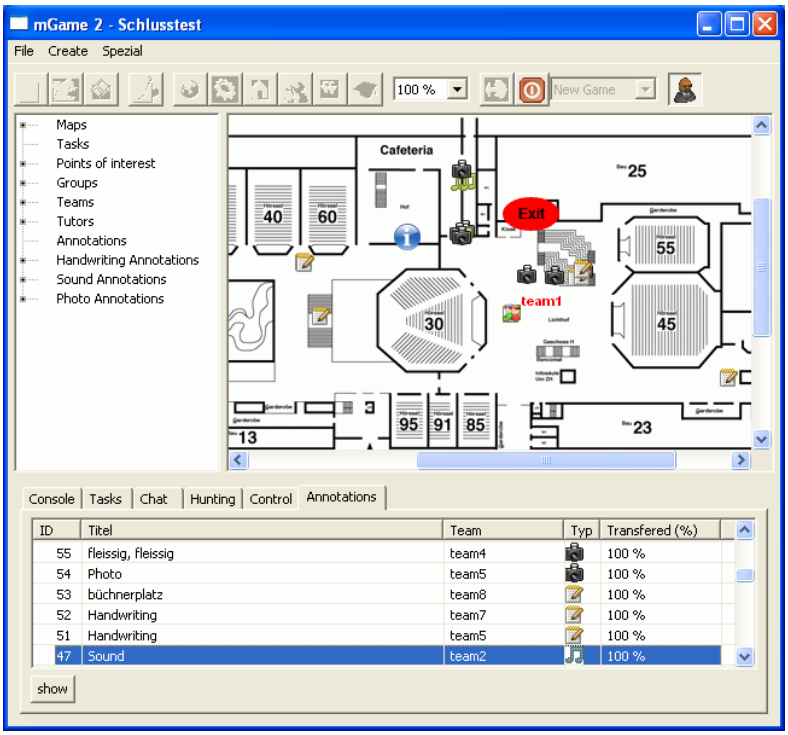
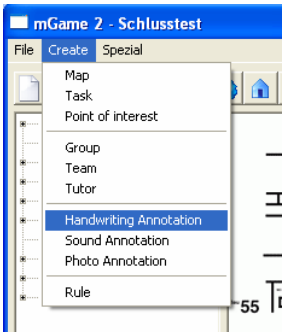


Abbildung 39 Server – Hauptansicht

Die Spalte „Typ“ besagt, um welche Annotationsart es sich handelt. Verschiedene Annotationen werden mit jeweils verschiedenen Symbolen gekennzeichnet, die auch auf der Karte ersichtlich sind. Wenn der Server nicht gestartet ist, können die einzelnen Annotations-Symbole frei auf der Karte verschoben und so die Position verändert werden.

5.2.2. Erstellen einer Handschrift-Annotation

Das Erstellen von Annotationen ist nur dann möglich, wenn der Server nicht gestartet ist. Um eine Handschrift-Annotation zu erstellen, wählt man im Menü „Create“ den Menüpunkt „Handwriting Annotation“ aus.



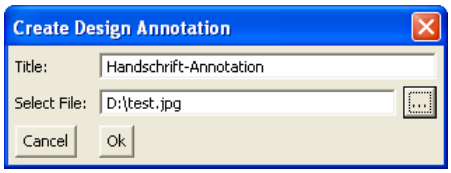


Abbildung 40 Server – Erstellen einer Handschrift-Annotation

Abbildung 40 zeigt den Dialog zur Erstellung der Handschrift-Annotation an. In diesem Dialog kann man den Titel der Annotation sowie eine Bilddatei angeben. Nach Bestätigung des Dialoges mit „Ok“ wird die Handschrift-Annotation auf der Karte mit einem entsprechenden Symbol dargestellt. Die Annotation ist nun erstellt und wird beim Start des Servers an alle Clients weitergeleitet, die am Mobile Game teilnehmen.

5.2.3. Erstellen einer Voice-Annotation

Nach der Auswahl des entsprechenden Menüeintrags im Menü „Create“ erscheint der Dialog zur Erstellung der Voice-Annotation. Analog zum Erstellen von Handschrift-Annotationen wird hier der Titel und der Pfad zu einer Datei angegeben. Die Datei muss im WAV-Format vorliegen.

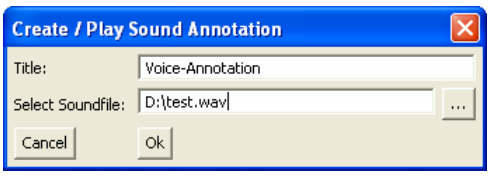


Abbildung 41 Server – Erstellen einer Voice-Annotation

Abbildung 41 zeigt den Dialog zum Erfassen von Voice-Annotationen.

5.2.4. Erstellen einer Foto-Annotation

Das Erfassen von Foto-Annotation funktioniert analog zum Erfassen von Handschrift-Annotationen in Kapitel 5.2.2. Nach erfolgreichem Erstellen wird aber das Symbol für die Voice-Annotationen auf der Karte angezeigt.

5.2.5. Entfernen von Annotationen

Annotationen können auch wieder entfernt werden. Dazu muss die zu löschende Annotation in der Baumstruktur ausgewählt werden. Durch einen Klick mit der rechten Maustaste erscheint ein Menü, welches über einen „Delete“-Eintrag verfügt. In Abbildung 42 ist dieser Vorgang dargestellt. Durch einen Klick auf den Eintrag wird die Annotation vom Server gelöscht.

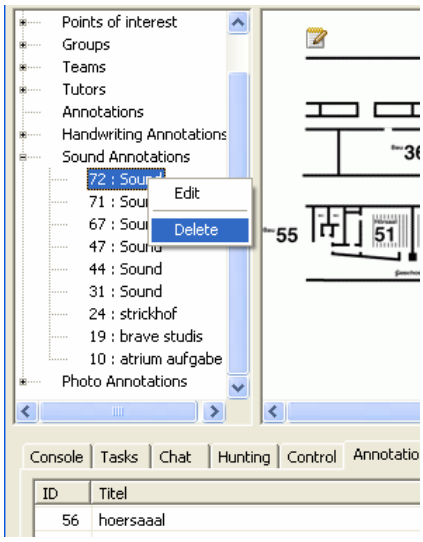


Abbildung 42 Server – Entfernen von Annotation

5.2.6. Anzeigen von Annotationen

Annotationen können auf verschiedene Arten angezeigt werden. Die erste Möglichkeit besteht darin, dass man in der Baumstruktur aus Abbildung 42 den Menüeintrag „Edit“ wählt. Ein ähnliches Menü erscheint, wenn man auf der Karte mit der rechten Maustaste auf ein Annotations-Symbol klickt. Abbildung 43 zeigt die Karte und das entsprechende Menü.

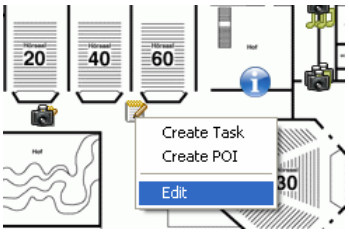


Abbildung 43 Server – Anzeigen von Annotationen 1

Eine weitere Möglichkeit zum Anzeigen besteht darin, auf der Lasche „Annotations“ die Schaltfläche „show“ zu betätigen. Dadurch wird die ausgewählte Annotation angezeigt.

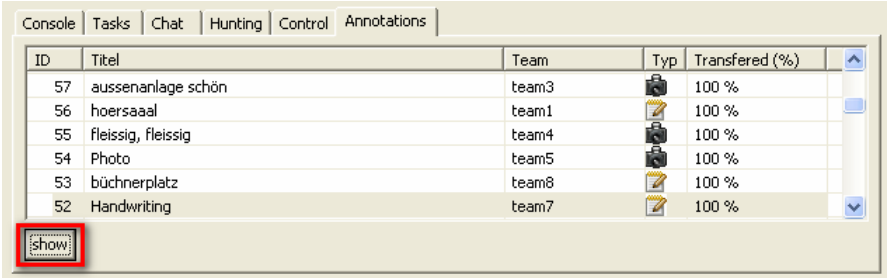


Abbildung 44 Server – Anzeigen von Annotationen 2

Abbildung 45 zeigt auf der linken Seite den Anzeigedialog für Handschrift- und Foto-Annotationen. Auf der rechten Seite ist der Dialog für Voice-Annotationen ersichtlich. Beim Betätigen der „Play“-Schaltfläche wird versucht, den Windows Media Player zu starten, um die Audiodaten wiederzugeben. Beide Anzeigedialoge verfügen über die Möglichkeit, die Daten der Annotation in Dateien zu speichern. Durch Klicken auf die „reply“-Schaltfläche werden die Annotationen zu der entsprechenden Annotation angezeigt.

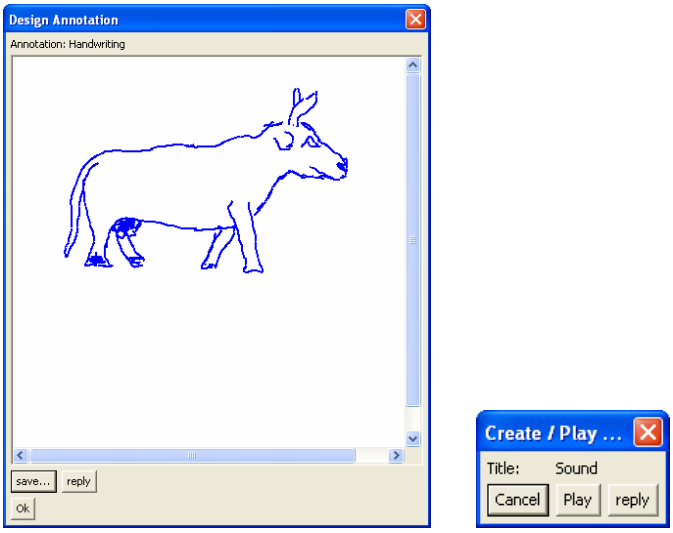


Abbildung 45 Server – Anzeigen von Annotationen 3

5.2.7. Exportieren von Annotationen

In der Hauptansicht kann man unter dem Menü „Spezial“ den Menüeintrag „Annotation-Export“ auswählen. Dann erscheint ein Dialog zum Auswählen eines Verzeichnisses. Wenn man ein gültiges Verzeichnis angegeben hat, kann der Export durchgeführt werden. Dieser speichert jede im aktuell geladenen Spiel vorhandene Annotation in einer Datei im angegebenen Verzeichnis ab.

6. Tests

Teil dieser Diplomarbeit sind drei Benutzertests, um die Funktionalität der entwickelten Applikation sicherzustellen. Der erste Test ist der so genannte Technikertest, bei dem es darum geht festzustellen, ob das entwickelte System auch im Zusammenspiel mit dem Gesamtsystem und der Infrastruktur funktioniert. Das heisst unter anderem auch, ob die Implementierung korrekt mit Ekahau[12] zusammenarbeitet. In einem zweiten Test wurde das Zusammenspiel mit mehreren Benutzern geprüft. Der dritte und letzte Test simulierte eine reale Spielumgebung.

6.1. Technikertest

Der Technikertest hat am 08.02.2007 stattgefunden und wurde mit zwei Personen durchgeführt, welche jeweils mit einem Qtek 9000 Gerät ausgestattet waren.

6.1.1. Aufgabe

Zusammen mit Christoph Göth wurde die grundsätzliche Funktionstauglichkeit der Annotationen im Mobile Game getestet. Ziel dieses Tests war, Annotation zu testen und gleichzeitig auch die Zusammenarbeit mit den mobilen Geräten, dem Positionierungssystem und dem Mobile Game Server zu verifizieren.

6.1.2. Durchführung

Anfänglich gab es einige Probleme, die jedoch mit den Gegebenheiten der drahtlosen Infrastuktur am IFI[13] und zum anderen mit Problemen der Konfiguration des Mobile Game Servers zusammenhingen. Diese konnten jedoch relativ schnell gelöst werden.
Während des Tests wurden Voice-, Foto-, Handschrift-, und Text-Annotationen erstellt und angezeigt.

6.1.3. Analyse

Der Test lief aus technischer Sicht ohne Probleme ab. Sowohl das Erstellen von Annotationen als auch das Anzeigen funktionierte einwandfrei. Es gab jedoch einige kleinere

[12] Ekahau ist das im Mobile Game implementierte System zur Positionsbestimmung

[13] Institut für Informatik

Anpassungsvorschläge. So wurde zum Beispiel von Herrn Schwabe vorgeschlagen, dass man alle Annotationen auch mit Sprachnachrichten annotieren könnte, nicht wie bisher nur mit Textnachrichten. Ebenfalls wünschte er eine Zoomfunktion der Fotovorschau bei Handschrift- und Foto-Annotationen. Des Weiteren wurde die Benutzerführung beim Erstellen von Voice-Annotationen bemängelt. Zusätzlich gab es noch ein Problem beim Anzeigen von Foto-Annotationen, da diese sehr lange Ladezeiten hatten. Dem Benutzer wurde aber keine entsprechende Meldung angezeigt und so hatte er den Eindruck, dass das Programm nicht mehr reagieren würde.

6.1.4. Fazit

Auf Grund des Tests wurde die Zoomfunktion für Bilder implementiert. Diese ermöglicht es nun, Bilder im Kleinformat anzuzeigen, so dass das ganze Bild auf dem Display des PDAs ersichtlich ist.

Die Benutzerführung für Voice-Annotationen wurde angepasst und entspricht nun derjenigen der Foto-Annotationen. Dadurch wird es für den Benutzer einfacher, Voice-Annotationen zu erstellen.

Beim Anzeigen von Foto-Annotationen wurde eine Statusmeldung implementiert, die besagt, dass das entsprechende Bild gerade geladen wird. Dies führt nicht mehr zu Missverständnissen auf grund der grösseren Ladezeiten von Bildern.

Das Annotieren einer Annotation mit Sprachnachrichten würde den Rahmen dieser Diplomarbeit sprengen, da grössere Umstrukturierungen der Datenschicht nötig wären.

6.2. Benutzertest

Der Benutzertest wurde am 15.02.2007 mit sechs Personen am IFI in Oerlikon durchgeführt. Zum Einsatz kamen dabei sechs Qtek 9000 Geräte.

6.2.1. Aufgabe

Aufgabe des Benutzertests war, die Funktionalität im Zusammenhang mit mehreren Spielern zu testen. Zusätzlich sollten die, aufgrund des Technikertests durchgeführten, Änderungen getestet werden. Zudem sollte den Teilnehmern am Ende des Tests ein Fragebogen verteilt werden, um die subjektiven Erfahrungen mit den Annotationen zu ermitteln. Die genaue Aufgabenbeschreibung befindet sich im Anhang B.

6.2.2. Durchführung

Zu Beginn des Tests gab es, wie auch bereits beim Technikertest, grössere Probleme mit der Ekahau Umgebung. Diesmal war der Rechner, auf dem der Mobile Game Server laufen sollte, nicht in der Lage, die Ekahau Tags zu erkennen. Deshalb musste der Server kurzfristig auf ein anderes System portiert werden. Nach dieser Umstellung lief der Test aus technischer Sicht ohne nennenswerte Probleme ab.

6.2.3. Analyse

Es gab auf den mobilen Geräten vereinzelte Programmunterbrüche, die aber laut Christoph Göth auf unlösbare Netzwerk-Infrastrukturprobleme zurückzuführen waren. So kam es beispielsweise vor, dass der PDA das drahtlose Netzwerk wechselte (vom research-Netz ins public-Netz). Da das System aber für das research-Netz konfiguriert wurde, war der Betrieb in der public-Umgebung nicht möglich. Dies führte dann zu unfreiwilligen Unterbrüchen und man musste manuell das Netzwerk wechseln und die Applikation auf dem mobilen Gerät neu starten. Andere technischen Probleme sind aber nicht aufgetaucht.

Durch den Test wurden jedoch kleinere Fehler in der Applikation aufgedeckt. So gab es zum Bespiel bei der Bildvorschau Probleme mit den Bildlaufleisten.

Von einer Testperson wurde die Bedienung des Spiels bemängelt. Die Person machte darauf aufmerksam, dass es enorm viele Klicks benötigt, um eine spezielle Aktion durchzuführen. Dazu zählte auch das Erstellen von Annotationen. Vor allem fand er das zwingende Erfassen des Titels pro Annotation mühsam. Die Testperson wünschte sich, dass bei der Foto-Annotation einfach nur der Auslöser gedrückt werden kann und dass dann die Annotation automatisch erstellt würde.

Die Auswertung der Fragebogen hat ergeben, dass alle Teilnehmer Freude am Spiel hatten. Alle empfanden das Spiel als hilfreich, um den Campus besser kennen zu lernen. Bemängelt wurden jedoch die Bedienung des PDAs sowie des Spiels an sich. Das Anzeigen von Annotationen bereitete den Benutzern keine Probleme, dagegen gab es vereinzelte Schwierigkeiten beim Erstellen. Handschrift- und Foto-Annotationen waren beliebt, Voice-Annotationen wurden hingegen oft als peinlich empfunden.

6.2.4. Fazit

Die Probleme mit der Bildvorschau wurden behoben und für den Schlusstest wurden die Änderungen ins System integriert.

Die Anregung betreffend den Anzahl Mausklicks konnte insofern umgesetzt werden, als man nun für die Annotationen einen Titel vorgibt. Dieser ist markiert und kann dann geändert werden, wenn dies gewünscht wird. Den Titel wegzulassen kam laut Christoph Göth nicht in Frage, da dieser ein zentraler Bestandteil der Annotation an sich ist. Das direkte Abfangen der Funktionstasten des PDAs (dazu zählt unter anderem die Fotoauslösungstaste) ist mit Java so nicht möglich. Dazu müsste man eine andere Systemumgebung definieren oder den Kameradialog ganz auslagern.

Die Analyse der Fragebogen hat ergeben, dass keine genaueren Aussagen über die einzelnen Annotationsarten gemacht werden konnten. Deshalb wurde entschieden, für den Schlusstest einen anderen Fragebogen zu verwenden. Zudem wurde ein zweiter Fragebogen erstellt, der vor dem Test ausgefüllt werden sollte. Dadurch sollte sich ermitteln lassen, was die Personen über Annotationen denken, bevor sie das Mobile Game gespielt haben und inwiefern sich diese Meinung durch das Spiel verändern würde.

6.3. Schlusstest

Der Schlusstest wurde am 23.02.2007 mit 15 Personen an der Uni Irchel durchgeführt. Zur Verfügung standen 8 mobile Endgeräte. Davon waren sechs Qtek 9000 und 2 HP iPAQ.

6.3.1. Aufgabe

Aufgabe des Schlusstests war es, die Funktionalität des Mobile Games mit vielen Spielern zu testen. Zu diesem Zweck mussten die Testpersonen verschiedene Aufgaben im Zusammenhang mit Annotationen ausführen. Eine genaue Beschreibung der Aufgabenstellung befindet sich im Anhang E.

6.3.2. Durchführung

Die Personen wurden in sieben Zweiergruppen und eine Einzelperson eingeteilt.

Nach einer kurzen Einführung wurde der erste Fragebogen verteilt. Danach begann der eigentliche Test, der ohne grössere Probleme durchgeführt werden konnte. Es gab aber wiederum vereinzelte Probleme mit dem Netzwerk.

6.3.3. Analyse

Der Test verlief aus technischer Sicht ohne Zwischenfälle. Vereinzelte Teams hatten Probleme mit dem Netzwerk, so dass sie die Applikation neu starten mussten.
In Tabelle 5 ist die Anzahl der erstellten Annotationen im Spiel dargestellt. Diese zeigt, dass Handschrift- und Foto-Annotationen am häufigsten erstellt wurden. Die Ergebnisse aus Tabelle 5 decken sich mit dem Resultat der Frage „Welche Annotationsart hat Ihnen am besten gefallen?“ des zweiten Fragebogens. Die beliebten Annotationen wurden also tatsächlich vermehrt verwendet.

Annotationsart	Anzahl erstellter Annotationen im Spiel
Handschrift-Annotation	16
Voice-Annotation	7
Foto-Annotation	16
Text-Annotation	0

Tabelle 5 Auswertung Annotationen

Wenn man die Frage „Welche Annotationsart denken Sie, gefällt Ihnen am besten?“ des ersten Fragebogens auswertet, erkennt man, dass die Text- und Foto-Annotation an erster Stelle liegen. Nur jeweils eine Person denkt, dass Ihr die Handschrift- und Voice-Annotation gefallen wird. Vergleicht man dieses Ergebnis mit der entsprechenden Frage des zweiten Fragebogens, wird deutlich, dass sich die Einstellungen der Personen während des Spiels geändert haben. So hat die Text-Annotation an Beliebtheit verloren (vor dem Spiel stimmten fünf Personen für die Text-Annotation, nach dem Spiel nur noch eine), und die Handschrift-Annotation gewonnen (vor dem Spiel stimmte eine Person für die Handschrift-Annotation, nach dem Spiel waren es sechs). Die Foto-Annotation war schon nach der ersten Auswertung mit sieben Stimmen an erster Stelle und erreichte sogar neun Stimmen, nachdem das Mobile Game gespielt wurde.
Bei der Frage „Welche Annotationsart wird für andere Studenten am hilfreichsten sein?“ zeigten sich weniger grosse Differenzen. Vor dem Spiel erhielt die Text-Annotation vier, die Handschrift-Annotation einen, die Voice-Annotation einen und die

Foto-Annotation acht Zähler. Nach dem Spiel waren es für die Text-Annotation nur noch drei, für die Handschrift-Annotation keine, für die Voice-Annotation zwei und für die Foto-Annotation elf Zähler. Trotz der grossen Beliebtheit der Handschrift-Annotation empfanden die Testpersonen den Nutzen gering und betrachteten die Handschrift-Annotation eher als Spassfaktor.

Frage	**Durchschnittswert (1 bis 5)**
Wie hat Ihnen das Spiel gefallen?	4.27
Wie hilfreich war das Spiel, um den Campus besser kennen zu lernen?	3.53
Wie gut konnten Sie den PDA bedienen?	3.40
Wie fanden Sie die Bedienung des Spiels?	3.20
Hatten Sie Probleme beim Erstellen von Annotationen?	2.33
Hatten Sie Probleme beim Anzeigen von Annotationen?	2.20
Wie gut hat Ihnen die Text-Annotation gefallen?	3.53
Wie gut hat Ihnen die Handschrift-Annotation gefallen?	3.73
Wie gut hat Ihnen die Voice-Annotation gefallen?	3.87
Wie gut hat Ihnen die Foto-Annotation gefallen?	4.53

Tabelle 6 Auswertung Fragen

Tabelle 6 zeigt die Auswertung der einzelnen Fragen. Dabei kann der Durchschnittswert einen Mindestwert von 1 und einen Höchstwert von 5 annehmen. Die Tabelle zeigt, dass das Spiel sehr gut aufgenommen wurde und den meisten Spass gemacht hat. Die Mehrheit der Testpersonen fand auch, dass sich das Spiel eignet, um den Campus besser kennen zu lernen. Die Bedienung des PDA und des Spiels sind mit einem Werten von 3.40 und 3.20 eher tief, liegen aber trotzdem noch über dem Durchschnitt von 2.5.
Das Erstellen und Anzeigen schien den Testpersonen keine grösseren Schwierigkeiten zu bereiten. Es ist ersichtlich, dass das Anzeigen einfacher war als das Erstellen, was ja auch durchaus verständlich ist.

Am wenigsten gefallen hat den Personen die Text-Annotation, die mit 3.53 Punkten am tiefsten bewertet wurde. Darauf folgt die Handschrift-Annotation mit 3.73 Zählern. Interessant ist hierbei, dass die Voice-Annotation mit 3.87 Punkten vor der Handschrift-Annotation liegt, obwohl bei der Frage „Welche Annotationsart hat Ihnen am besten gefallen?“ niemand für die Voice-Annotation gestimmt hat. An der Spitze liegt mit einigem Abstand die Foto-Annotation mit einem Wert von 4.53.

Eine genauere Betrachtung der einzelnen Annotationen hat ergeben, dass die Qualität bei den Voice-Annotationen am besten ist. Die Aufzeichnungen wurden mit grosser Sorgfalt durchgeführt und der Inhalt ist klar verständlich. Vereinzelt waren die Aufnahmen etwas zu leise, was auf die unsachgemässe Anwendung der Sprachaufzeichnungskomponente des mobilen Gerätes zurückzuführen ist. Bei den Foto-Annotationen gab es Probleme mit der Qualität der Bilder. Oft waren sie unterbelichtet oder unscharf. Dies hat aber mit der Handhabung der integrierten Kamera des Qtek-Gerätes zu tun. Die Handschrift-Annotationen waren am kreativsten und am vielseitigsten. Sie zeigten jedoch nur eine geringe Aussagekraft und dienten eher dem Spassfaktor.

6.3.4. Fazit

Das Spiel hat den Testpersonen gut gefallen. Vereinzelt gab es Probleme mit der Netzwerkinfrastruktur. Diese führten zu Problemen innerhalb des Spiels. Eine Analyse der Logdateien des Spiels und der Ergebnisse der Fragebogen hat gezeigt, dass die Personen, welche während des Spiels mit technischen Problemen zu kämpfen hatten, sowohl Abzüge in den Bewertungen für die Bedienung des Spiels als auch für das Erstellen und Anzeigen von Annotationen machten.

Die Analyse der Fragebogen zeigte, dass grossen Wert darauf gelegt wurde, dass die Annotationen schnell gemacht werden konnten. Zudem wurde erwähnt, dass Fotos universal verständlich wären und deshalb bevorzugt werden sollten.

7. Fazit

Die in dieser Arbeit betrachtete Erweiterung des Mobile Games zur Unterstützung von Handschrift-, Voice- und Foto-Annotationen konnte erfolgreich implementiert werden. Die korrekte Funktionalität wurde durch verschiedene Benutzertests bestätigt.

Die neuen Annotationen stellen für das Mobile Game einen Mehrwert dar und werden von den Benutzern gerne verwendet. Die Benutzertests haben gezeigt, dass es keine grossen Schwierigkeiten beim Erstellen und Anzeigen von Annotationen gibt.

Bemängelt wurde teilweise die schlechte Fotoqualität oder das mühsame Schreiben mit dem Stift auf dem PDA. Diese Probleme können jedoch nicht im Zusammenhang mit dem Mobile Game gelöst werden und sind Gegenstand der Hardware-Entwicklung dieser mobilen Geräte.

7.1. Weiterführende Arbeiten

In diesem Kapitel geht es darum aufzuzeigen, welche zukünftigen Forschungen auf dieser Arbeit aufbauen könnten.

Netzwerkprobleme und Benutzerfreundlichkeit

Während den Benutzertests wurden immer wieder grössere Probleme mit dem Netzwerk festgestellt. Diese waren auf Netzwerkunterbrüche oder unkontrollierten Netzwerkwechsel zurückzuführen. Dies führte meistens dazu, dass das Mobile Game nicht mehr richtig funktionierte und neu gestartet werden musste. Hier könnte man versuchen, auf Applikationsebene besser auf entsprechende Gegebenheiten zu reagieren. So wäre es zum Beispiel sinnvoll, dem Benutzer mitzuteilen, dass er keinen Empfang hat oder dass er sich im falschen Netzwerk befindet.

Zusätzliche Benutzertests

Um den Mehrwert der Annotationen besser aufzuzeigen, wären weitere Benutzertests sinnvoll, in welchen gezielt mit Annotationen gearbeitet würden. Man könnte somit auch feststellen, ob es sinnvoll wäre, alle Annotationen bereitzustellen oder nicht. So könnte man zum Beispiel herausfinden, ob die Handschrift-Annotation nur witzig für den Ersteller ist oder ob sie auch effektiv zu einem Mehrwert führen kann.

Erweiterte Foto-Annotation

Einige Personen aus den Benutzertests haben angemerkt, dass die Annotation durch Fotos sehr sinnvoll, aber nicht vollständig ist. Die Idee war, einzelne Bereiche eines Fotos mit zusätzlichem Text zu versehen, um noch genauere Informationen bereitzustellen. In Flickr ist dies teilweise umgesetzt. Zudem wäre es auch denkbar, die bisherigen Annotationen nicht nur mit Text, sondern auch mit Handschrift, Voice oder Foto zu annotieren.

Verbesserte Bedienung

Die Benutzertests haben gezeigt, dass im Bereich der Bedienung des PDAs und des Mobile Games noch viel Verbesserungspotential besteht. Hier wäre eine grundlegende Analyse dieser Materie sinnvoll.

Entwicklungsumgebung

Ein weiterer Punkt ist die Überlegung, ob für die Zukunft nicht eine andere Entwicklungsumgebung gewählt werden sollte. Das Mobile Game wurde mit Java 2 Micro Edition (J2ME) entwickelt und basiert auf der J9-Runtime. Im Rahmen dieser Arbeit wurde festgestellt, dass viele Funktionen ausgelagert werden mussten, da diese durch die Entwicklungsumgebung nicht unterstützt wurden. Dazu gehört zum Beispiel das Ansteuern der Fotokamera und der Audioaufzeichnung. Es stellt sich die Frage, ob eine Entwicklung mit dem .Net Compact Framework nicht sinnvoller wäre, da man dann einfacher externe Bibliotheken einbinden und mobile Applikationen leichter entwickeln könnte.

7.2. Persönliches Schlusswort

Durch diese Arbeit habe ich viele interessante Aspekte der Entwicklung mobiler Geräte kennen gelernt. So wurde mir unter anderem bewusst, dass eine Applikation immer unter den realen Bedingungen getestet werden muss, um festzustellen, ob das Verhalten dem Verhalten in der Entwicklungsumgebung entspricht. Gerade bei der Netzwerkinfrastruktur war die Funktionsweise oftmals sehr schwer vorherzusagen.

Besonderer Dank verdient mein Betreuer Christoph Göth, der mich während der ganzen Arbeit unterstützt hat und bei den Benutzertests immer mit grossem Einsatz Hilfe leistete.

8. Verwendete Quellen

8.1. Literatur

[BeBr2002] Bernheim Brush, A. J. Annotating Digital Documents for Asynchronous Collaboration, 2002, Washington, University of Washington

[Brunner] Brunner, D. Architekturentwicklung für mobile Spiele, 2005, Diplomarbeit, Universität Zürich, Institut für Informatik.

[Espinoza] Espinoza, Fredrik et all. GeoNotes: Social and Navigational Aspects of Location Based Information Systems, HUMLE Lab, Swedish Institute of Computer Science (SICS)

[FroSchw2004] Frohberg, Dirk. Schwabe, Gerhard. Der mobile Mehrwert von Annotationen im mCSCL, 2004, Universität Zürich[Göth2003] Göth, Christoph. Prototypische Implementierung einer mobilen Spieleumgebung für den PDA, Diplomarbeit, 2003 Universität Koblenz

[Kienle2003] Kienle, A. Integration von Wissensmanagement und kollaborativem Lernen durch technisch unterstützte Kommunikationsprozesse, 2003, Köln

[Nova2005] Nova, N et all. A Mobile Game to Explore the Use of Location Awareness on Collaboration, 2005, EPFL Lausanne

[Persson2002] Persson, Per et all. GeoNotes: A Location-based Information System for Public Spaces, 2002, Swedish Institute of Computer Science (SICS)

[Sarvas2004] Sarvas, R., Herrarte, E., Wilhelm, A., Davis, M. Metadata Creation System for Mobile Images, 2004, ACM Press

[Suter] Suter, Daniel. Indoornavigation unterstützt durch Magnetfeld-sensorik, Diplomarbeit, 2006, Universität Zürich, Institut für Informatik

[Schiller2003] Schiller Jochen. Mobilkommunikation. 2. Auflage 2003 Pearson Studium München

[Tanenbaum2003] Tanenbaum Andrew S. Computernetzwerke. 4. Auflage. 2003 Pearson Studium München

[Wilhelm2004] Wilhelm, Anita et all. Photo Annotation on a Camera Phone, 2004, ACM Press

8.2. Online Quellen

[CatchBob] http://craftwww.epfl.ch/research/catchbob/
(Zuletzt aufgerufen am 04.03.2007)

[Flickr] http://www.flickr.com
(Zuletzt aufgerufen am 04.03.2007)

[GeoNotes] http://geonotes.sics.se/
(Zuletzt aufgerufen am 04.03.2007)

[IBMWebSphere] http://www-306.ibm.com/software/wireless/wsdd/
(Zuletzt aufgerufen am 04.03.2007)

[IBMJ9] http://www-306.ibm.com/software/wireless/weme/
(Zuletzt aufgerufen am 04.03.2007)

[JavaSerialize] http://java.sun.com/j2se/1.5.0/docs/guide/serialization/spec/serialTOC.html
(Zuletzt aufgerufen am 04.03.2007)

[SWTToolkit] http://www.eclipse.org/swt/
(Zuletzt aufgerufen am 04.03.2007)

9. Verzeichnisse

9.1. Abbildungsverzeichnis

9.2. Tabellenverzeichnis

9.3. Quellcodeverzeichnis

9.4. Abkürzungen

ACK	ACKnowledgement
BLOB	Binary Large Object
DLL	Dynamic Link Library
GSM	Global System for Mobile Communication
GPRS	General Packet Radio Service
IEEE	Institute of Electrical and Electronics Engineers
IFI	Institut für Informatik
IP	Internet Protocol
J2ME	Java 2 Micro Edition
JNI	Java Native Interface
JPEG	Joint Photographics Expert Group
LAN	Local Area Network
GPS	Global Positioning System
GUI	Graphical User Interface
PDA	Personal Digital Assistant
SWT	Standard Widget Toolkit
TCP	Transmission Control Protocoll
WLAN	Wireless Local Area Network

Anhang A: Aufgabenstellung

Im Rahmen des Forschungsgebietes „Mobile Learning Games“ hat Christoph Göth eine interaktive und mobile Spieleumgebung entwickelt. In der nun zu bearbeitenden Diplomarbeit geht es darum, Annotationen zu implementieren. Dabei handelt es sich um Informationen, die von einem Spieler an einer beliebigen Stelle bereitgestellt werden können und diese dann für andere Personen ersichtlich sein sollen. Diese Annotationen werden in folgende drei Bereiche unterteilt:

- **Foto-Annotationen**
  Fotos sollen an beliebigen Stellen im Spiel gemacht werden können und sollen für andere Spieler ersichtlich sein.
- **Voice-Annotationen**
  Sprachaufzeichnungen sollen mit Hilfe der in den entsprechenden PDAs integrierten Mikrophonen aufgenommen und anschliessend den anderen Spielern zu Verfügung gestellt werden.
- **Handschrift/Zeichnungen**
  Mittels PDA-Stift auf dem Touchscreen erstellte Zeichnungen bzw. handgeschriebene Notizen sollen anderen Spielern zur Verfügung gestellt werden.

Diese Arbeit besteht aus einem Literatur- und Implementierungsteil. Im Theorieteil sollen bisherige Entwicklungen im Bereich von Annotationen recherchiert und analysiert werden.

Im Implementationsteil sollen folgende Komponenten entwickelt werden:

- Ein serverseitiges Programm, welches folgende Aufgaben abdeckt:
  - ➔ Verwaltung der Ressourcen (Bilder / Sprachaufzeichnungen etc.)
  - ➔ Verteilung der Annotationsinformationen (Position) an andere Spieler
  - ➔ Übertragung der Daten an den Client
- Ein clientseitiges Programm, das folgende Bereiche abdeckt:
  - ➔ Erfassen von Annotationen (Erstellen von Fotos / Zeichnungen)
  - ➔ Übertragen der Daten an den Server
  - ➔ Anzeige von bereits vorhandenen Annotationen

Das entwickelte System soll mehrbenutzertauglich sein und soll auch mit unterschiedlichen Netzwerkbedingungen (Schwankende Signalstärken, Bewegungen des Clients, Netzunterbrüche) zu Recht kommen.

Das System soll dokumentiert und einer praktischen Evaluation unterzogen werden, wobei ca. 20 Personen in einer Testreihe das „Mobile Learning Game“ spielen. Mittels mehrerer Zwischentests während der Entwicklungsphase wird der Fortschritt der Arbeit sichergestellt.

Im Schlussteil der Arbeit soll das Erreichte dokumentiert und Folgerungen für zukünftige Arbeiten präsentiert werden.

Anhang B: Aufgabenstellung Benutzertest

Rahmenhandlung / Szenario

Sich an einer Uni zurechtzufinden ist für neue Studenten immer schwer. Diese Aufgaben sollen zeigen, dass man relevante Objekte auch spielerisch finden kann. Annotationen sollen dazu dienen, die gefundenen Orte zu vermerken.

Handschrift

- Finde Sie 5 interessante Objekte und erstellen Sie eine Zeichnung davon.
- Zeichnen Sie das aktuelle Tagesmenu der Mensa
- Erstellen Sie eine Karikatur Ihres Teamkollegen vor dem Haupteingang
    - Nehmen Sie die Reaktion Ihres Teamkollegen auf
- Lösen Sie die Aufgabe, welche sich an der Tür bei den Stundentenarbeitsplätzen befindet.

Voice

- Beschreiben Sie den Blick in die Mensa.
- Was sehen Sie, wenn Sie beim 2. Stock aus dem Lift steigen.
- Beschreiben Sie ein Inserat / Anzeige an der Pinwand beim Eingang
- Fragen Sie jemanden, wieso er sich an der Uni aufhält.
- Lösen Sie die Aufgabe, welche sich an der Tür bei der Bibliothek befindet.

Fotos

- Finden Sie 5 interessante Objekte und machen Sie ein Foto davon.
- Fotografieren Sie das aktuelle Tagesmenu der Mensa
- Fotografieren Sie den Blick aus einem Fenster im 2. Stock

Replys

- Kommentiere 3 Zeichnungen anderer Teams
- Kommentiere 3 Fotos anderer Teams
- Kommentiere 3 Voice-Nachrichten anderer Teams

Anhang C: Fragebogen Benutzertest

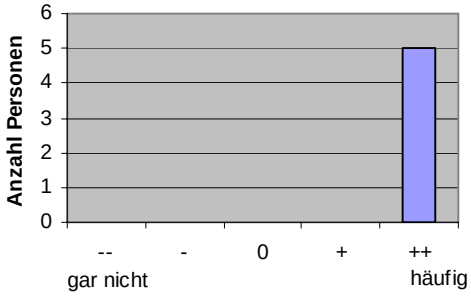
1	Teamname	
2	Geburtsjahr	
3	Geschlecht	männlich ☐ weiblich ☐
4	Verwenden Sie regelmässig Computer?	gar nicht -- - **0** + ++ häufig
5	Hatten Sie bereits vorher Erfahrungen mit PDA's gesammelt?	gar keine -- - **0** + ++ sehr gut
6	Wie hat Ihnen das Spiel gefallen?	gar nicht -- - **0** + ++ sehr gut
7	Wie gut kannten Sie die Universität vorher?	gar nicht -- - **0** + ++ sehr gut
8	Wie hilfreich war das Spiel, um den Campus besser kennen zu lernen?	gar nicht -- - **0** + ++ sehr viel
9	Wie gut konnten Sie den PDA bedienen?	gar nicht -- - **0** + ++ sehr gut
10	Wie fanden Sie die Bedienung des Spiels?	sehr schwierig -- - **0** + ++ sehr einfach
11	Hatten Sie Probleme beim Erstellen von Annotationen?	gar keine -- - **0** + ++ sehr viele
12	Hatten Sie Probleme beim Anzeigen von Annotationen?	gar keine -- - **0** + ++ sehr viele
13	Welche Annotationsart hat Ihnen am besten gefallen?	Handwriting-Annotation ☐ Voice-Annotation ☐ Foto-Annotation ☐ Begründung:

14 Beschreiben Sie kurz, was Ihnen besonders gut gefallen hat:

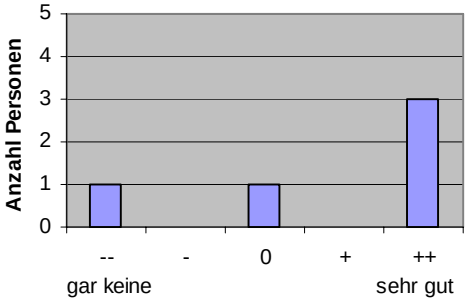
15 Beschreiben Sie kurz, was Ihnen gar nicht gefallen hat:

Anhang D: Auswertung Benutzertest

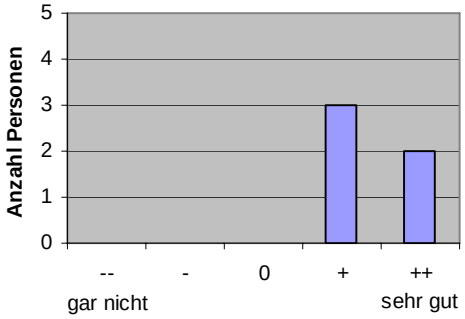
Verwenden Sie regelmässig Computer?



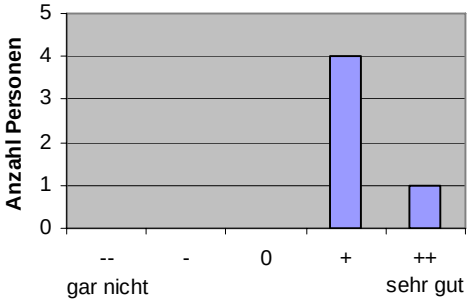
Hatten Sie bereits vorher Erfahrungen mit PDA's gesammelt?



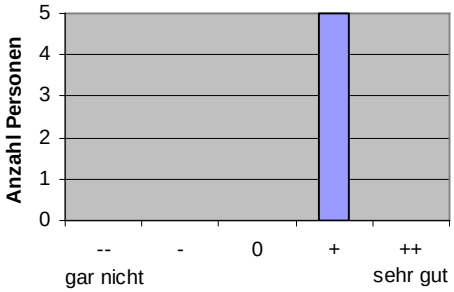
Wie hat Ihnen das Spiel gefallen?



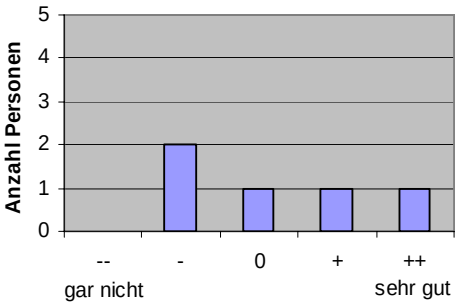
Wie gut kannten Sie die Universität vorher?



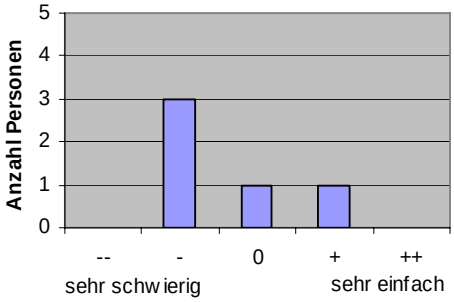
Wie hilfreich war das Spiel um den Campus besser kennen zu lernen?



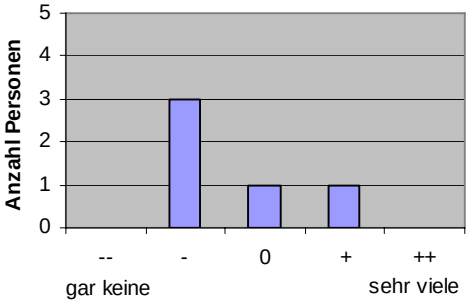
Wie gut konnten Sie den PDA bedienen?

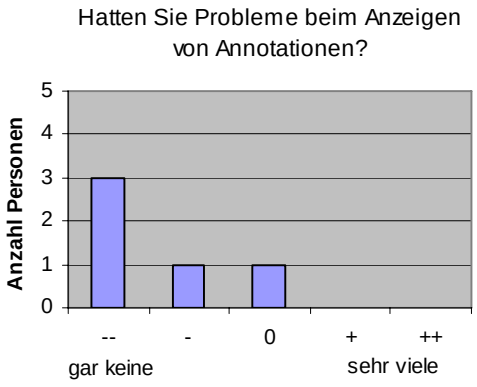


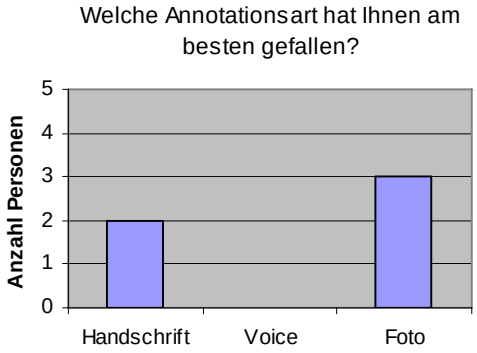
Wie fanden Sie die Bedienung des Spiels?



Hatten Sie Probleme beim Erstellen von Annotationen?




Hatten Sie Probleme beim Anzeigen von Annotationen?
Anzahl Personen
5
4
3
2
1
0
--
-
0
+
++
gar keine
sehr viele


Welche Annotationsart hat Ihnen am besten gefallen?
Anzahl Personen
5
4
3
2
1
0
Handschrift
Voice
Foto

Anhang E: Aufgabenstellung Schlusstest

Rahmenhandlung / Szenario

Sich an einer Uni zurechtzufinden ist für neue Studenten immer schwer. Wir werden ihnen dabei helfen. Aus diesem Grund verschönern wir den Campus mit Fotos und Zeichnungen und hinterlegen an interessanten Orten Sprachnachrichten. So können Studenten auch Orte kennen lernen, die sie selbst nicht besucht haben. Durch Annotationen werden die Studenten ermutigt auch Orte zu besuchen, die sie vielleicht ohne die zusätzlichen Anmerkungen nicht aufgesucht hätten.

Handschrift

- Finden Sie 3 interessante Objekte und erstellen Sie eine Zeichnung davon.
- Zeichnen Sie das aktuelle Tagesmenu der Mensa
- Erstellen Sie eine Karikatur Ihres Teamkollegen
    - Nehmen Sie die Reaktion Ihres Teamkollegen auf
- Lösen Sie die Aufgabe, welche sich an der Tür beim Hörsaal G45 befindet.

Voice

- Beschreiben Sie den Strickhof.
- Beschreiben Sie ein Inserat / Anzeige an der Pinwand beim Eingang
- Fragen Sie jemanden, wieso er sich an der Uni aufhält.
- Lösen Sie die Aufgabe, welche sich an der Tür beim Cafe Atrium (Grüne Cafeteria bei den Hörsälen) befindet.

Fotos

- Finden Sie 3 interessante Objekte und machen Sie ein Foto davon.
- Fotografieren Sie das Cafe Atrium (Grüne Cafeteria bei den Hörsälen)
- Fotografieren Sie den Blick auf die Aussenanlage

Replys

- Kommentiere 2 Zeichnungen anderer Teams
- Kommentiere 2 Fotos anderer Teams
- Kommentiere 2 Voice-Nachrichten anderer Teams

Anhang F: Fragebogen 1 Schlusstest

1	Teamname	
2	Geburtsjahr	
3	Geschlecht	männlich ☐ weiblich ☐
4	Verwenden Sie regelmässig Computer?	gar nicht -- - **0** + ++ häufig
5	Hatten Sie bereits vorher Erfahrungen mit PDA's gesammelt?	gar keine -- - **0** + ++ sehr gut
6	Wie gut kennen Sie den Campus?	gar nicht -- - **0** + ++ sehr gut

7	Welche Annotationsart denken Sie, gefällt Ihnen am besten?	Text-Annotation ☐ Handwriting-Annotation ☐ Voice-Annotation ☐ Foto-Annotation ☐ Begründung:
8	Welche Annotationsart wird für andere Stundenten am hilfreichsten sein?	Text-Annotation ☐ Handwriting-Annotation ☐ Voice-Annotation ☐ Foto-Annotation ☐ Begründung:

Anhang G: Fragebogen 2 Schlusstest

1	Teamname	

2	Wie hat Ihnen das Spiel gefallen?	gar nicht	--	-	**0**	+	++	sehr gut
3	Wie hilfreich war das Spiel, um den Campus besser kennen zu lernen?	gar nicht	--	-	**0**	+	++	sehr viel
4	Wie gut konnten Sie den PDA bedienen?	gar nicht	--	-	**0**	+	++	sehr gut

5	Wie fanden Sie die Bedienung des Spiels?	sehr schwierig	--	-	**0**	+	++	sehr einfach
6	Hatten Sie Probleme beim Erstellen von Annotationen?	gar keine	--	-	**0**	+	++	sehr viele
7	Hatten Sie Probleme beim Anzeigen von Annotationen?	gar keine	--	-	**0**	+	++	sehr viele
8	Wie gut hat Ihnen die Text-Annotation gefallen	gar nicht	--	-	**0**	+	++	sehr gut Begründung:
9	Wie gut hat Ihnen die Handschrift-Annotation gefallen	gar nicht	--	-	**0**	+	++	sehr gut Begründung:
10	Wie gut hat Ihnen die Voice-Annotation gefallen	gar nicht	--	-	**0**	+	++	sehr gut Begründung:

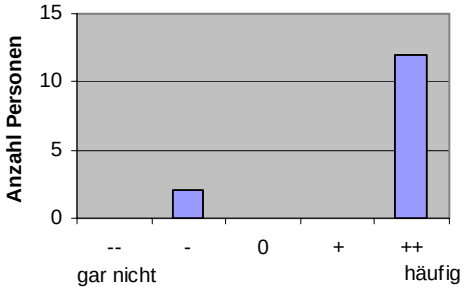
11	Wie gut hat Ihnen die Foto-Annotation gefallen	gar nicht ☐ -- ☐ - ☐ **0** ☐ + ☐ ++ ☐ sehr gut Begründung:
12	Welche Annotationsart hat Ihnen am besten gefallen?	Text-Annotation ☐ Handwriting-Annotation ☐ Voice-Annotation ☐ Foto-Annotation ☐ Begründung:
13	Welche Annotationsart wird für andere Stundenten am hilfreichsten sein?	Text-Annotation ☐ Handwriting-Annotation ☐ Voice-Annotation ☐ Foto-Annotation ☐ Begründung:

14 Beschreiben Sie kurz, was Ihnen besonders gut gefallen hat:

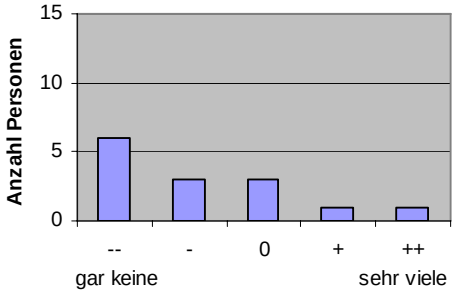
15 Beschreiben Sie kurz, was Ihnen gar nicht gefallen hat:

Anhang H: Auswertung Fragebogen 1 Schlusstest

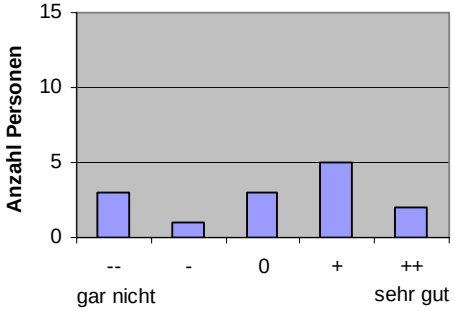
Verwenden Sie regelmässig Computer?



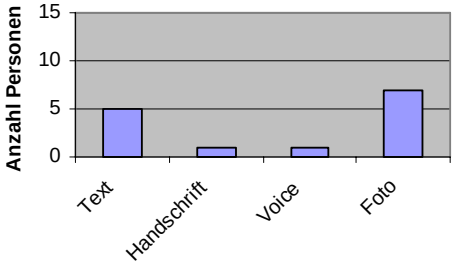
Hatten Sie bereits vorher Erfahrungen mit PDA's gesammelt?



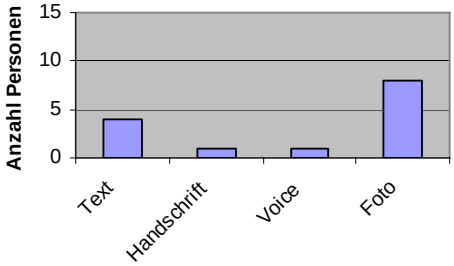
Wie gut kennen Sie den Campus?



Welche Annotationsart denken Sie, gefällt Ihnen am besten?

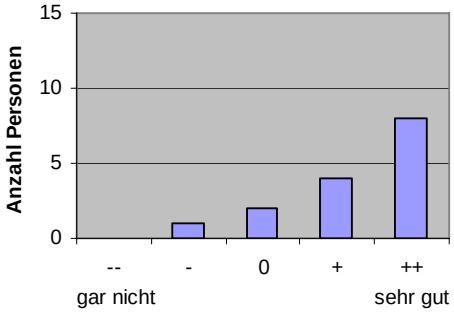


Welche Annotationsart wird für andere Stundeten am hilfreichsten sein?

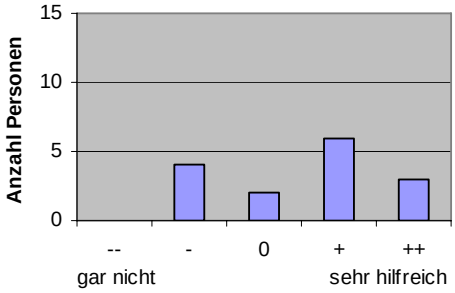


Anhang I: Auswertung Fragebogen 2 Schlusstest

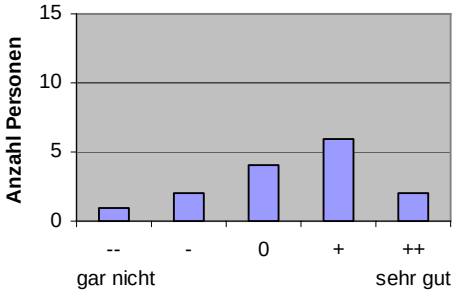
Wie hat Ihnen das Spiel gefallen?



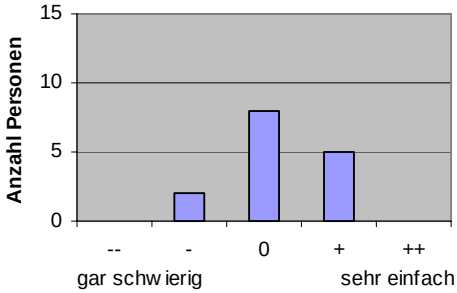
Wie hilfreich war das Spiel um den Campus besser kennen zu lernen?



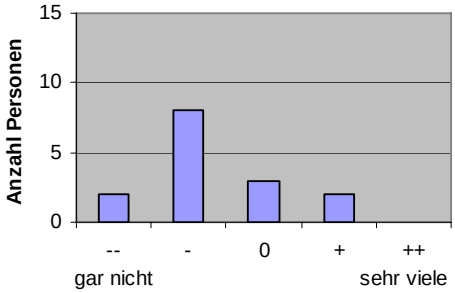
Wie gut konnten Sie den PDA bedienen?



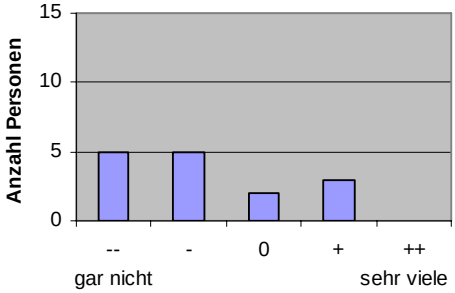
Wie fanden Sie die Bedienung des Spiels?



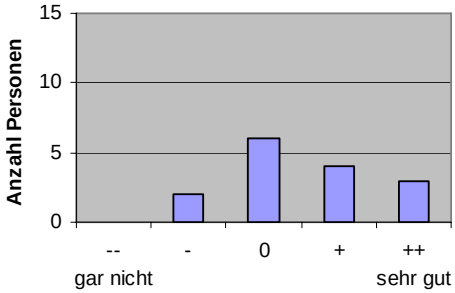
Hatten Sie Probleme beim Erstellen von Annotationen?



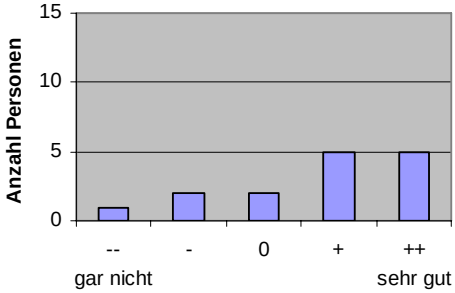
Hatten Sie Probleme beim Anzeigen von Annotationen?

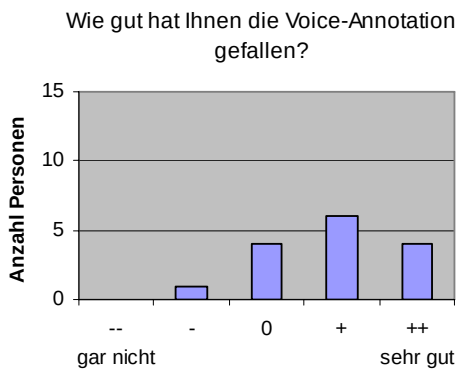


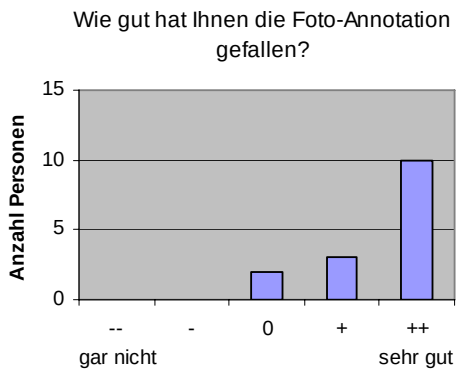
Wie gut hat Ihnen die Text-Annotation gefallen?

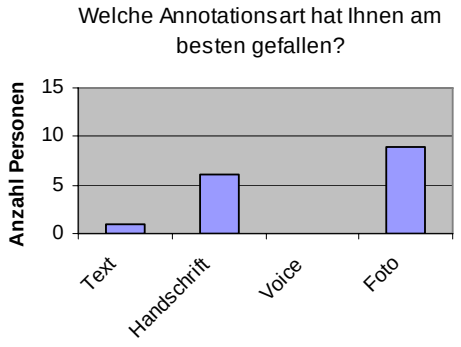


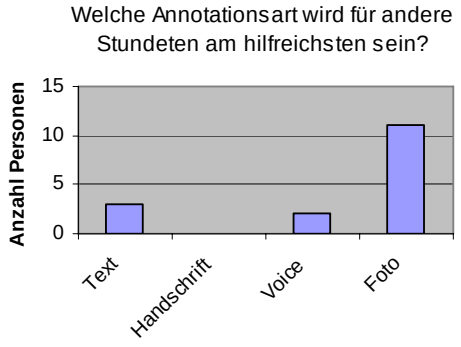
Wie gut hat Ihnen die Handschrift-Annotation gefallen?




Wie gut hat Ihnen die Voice-Annotation gefallen?
Anzahl Personen
15
10
5
0
--
-
0
+
++
gar nicht
sehr gut


Wie gut hat Ihnen die Foto-Annotation gefallen?
Anzahl Personen
15
10
5
0
--
-
0
+
++
gar nicht
sehr gut


Welche Annotationsart hat Ihnen am besten gefallen?
Anzahl Personen
15
10
5
0
Text
Handschrift
Voice
Foto


Welche Annotationsart wird für andere Stundeten am hilfreichsten sein?
Anzahl Personen
15
10
5
0
Text
Handschrift
Voice
Foto

Anhang J: Inhalt der CD-ROM

Zu dieser Diplomarbeit gehört ebenfalls eine CD-ROM und beinhaltet neben dieser Diplomarbeit als PDF-Datei auch den Quellcode des Mobile Games.

Verzeichnis	Inhalt
Diplomarbeit	Diplomarbeit als PDF-Datei Zusammenfassung Abstract
JavaDoc_Client	Dokumentation des Mobile Game Clients
JavaDoc_Server	Dokumentation des Mobile Game Servers
Mobile_Game_Client	Quellcode Mobile Game Client
Mobile_Game_Server	Quellcode Mobile Game Server
JNI	Quellcode c++ Bibliotheken
Technikertest	Quellcode zum Zeitpunkt des Tests
Benutzertest	Quellcode zum Zeitpunkt des Tests Ausgefüllte Fragebogen
Schlusstest	Quellcode zum Zeitpunkt des Tests Gespeichertes Spiel Exportierte Annotationen Präsentationsunterlagen Ausgefüllte Fragebogen Konfiguration und Logdateien

Tabelle 7 Inhalt der CD-ROM

Anhang K: CD-ROM