Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Diplomarbeit

Roman Bartnik

Weiterentwicklung einer Technologiebasis für interaktive Gruppenarbeitsräume

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Roman Bartnik

Weiterentwicklung einer Technologiebasis für interaktive Gruppenarbeitsräume

Diplomarbeit eingereicht im Rahmen der Diplomprüfung
im Studiengang Softwaretechnik
am Studiendepartment Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Kai von Luck
Zweitgutachter : Prof. Dr. rer. nat. Jörg Raasch

Abgegeben am 23.2.2006

Roman Bartnik

Thema der Diplomarbeit

Weiterentwicklung einer Technologiebasis für interaktive Gruppenarbeitsräume

Stichworte

Ubiquitous Computing, CSCW, Groupware, Mensch-Computer-Interaktion

Kurzzusammenfassung

Die Fülle an Informationen, denen ein Mensch ausgesetzt ist, wird auch in Zukunft weiter steigen. Insbesondere Teilnehmer von Arbeitstreffen müssen nicht nur mit diesen Informationen arbeiten, sondern auch die Kommunikation innerhalb einer Gruppe koordinieren. Daher sind in einem Konferenzraum die Werkzeuge zur Informationsbewältigung besonders wichtig. Wie Teilnehmer eines Meetings durch Einsatz vernetzter Geräte optimal unterstützt werden können, ist Thema der vorliegenden Arbeit. Die dafür notwendigen, im Folgenden erarbeiteten Anforderungen fließen in eine Architektur ein, die offen genug für spätere Erweiterungen ist. Diese Architektur wird realisiert und ihre Tauglichkeit anhand von zwei Anwendungsprototypen demonstriert.

Roman Bartnik

Title of the paper

Further development of a technology platform for interactive collaborative rooms

Keywords

Ubiquitous Computing, CSCW Groupware Human-Computer-Interaction

Abstract

The future holds a new wealth of information for everybody. People will need the proper tools to handle this wealth, or they will be overwhelmed by it. In a conference room this is of special importance, as they will not only have to manage the information, but also coordinate with coworkers. This work shows how participants of a meeting can be supported by the use of ubiquitous computing technology. Based on the previous findings an architecture open for extensions is developed and realised. Two application prototypes are used as a proof of concept for this implementation.

Inhaltsverzeichnis

Abbildungsverzeichnis **7**

Tabellenverzeichnis **8**

1. Einleitung **9**
1.1. Motivation . . . 9
1.2. Szenario . . . 11
1.2.1. Ablauf eines fiktiven Meetings . . . 11
1.2.2. Gezeigte Anwendungen . . . 12
1.3. Ziel der Arbeit . . . 13
1.4. Gliederung der Arbeit . . . 13

2. Analyse **14**
2.1. Rahmenbedingungen und Eigenschaften der Zielgruppe . . . 14
2.1.1. Zeit, Ort und Art der Gruppe . . . 14
2.1.2. Steuerung und Usability . . . 15
2.1.3. Systemannahmen . . . 16
2.2. Beschreibung des Status Quo . . . 16
2.3. Szenario (Soll-Zustand) . . . 18
2.3.1. Einfaches Mitteilen von Informationen . . . 18
2.3.2. Gemeinsames Bearbeiten . . . 19
2.3.3. Einfaches Festhalten von Ideen . . . 19
2.3.4. Weitere Forderungen . . . 19
2.4. Technische Anforderungen . . . 20
2.4.1. Benutzer – Geräte Konfigurationen . . . 21
2.4.2. Wissen um die Raumumgebung . . . 22
2.4.3. Objektpersistenz . . . 23
2.4.4. Darstellung . . . 24
2.4.5. Bearbeiten . . . 24
2.4.6. Verwaltung . . . 26
2.5. Anwendungsfälle . . . 27
2.5.1. Systemadministration . . . 27
2.5.2. Dienste nutzen . . . 28

2.5.3. Anzeige von Dokumenten 29
2.5.4. Geschichten 29
2.6. Zusammenfassung 30

3. Stand der Technik **32**
3.1. Verteilte Grafikausgabe 32
3.1.1. Remote Desktop 33
3.1.2. Supportsoftware 33
3.1.3. Remote Collaboration 34
3.1.4. Fazit 34
3.2. Öffentliche Bildschirme 34
3.2.1. Smartboards 35
3.2.2. BlueBoard 35
3.2.3. Fazit 37
3.3. Gemeinschaftliche Anwendungen 37
3.4. Gemeinsam genutzte Datenspeicher 39
3.5. Ubiquitous Computing Umgebungen 39
3.5.1. Roomware 40
3.5.2. Interactive Workspaces 42
3.5.3. Fazit 45
3.6. Zusammenfassung 46

4. Design **47**
4.1. Annahmen und Eingrenzungen 47
4.2. Grundlegendes 48
4.2.1. Außensicht 48
4.2.2. Aufteilungsmöglichkeiten 49
4.2.3. Zentraler Server oder verteiltes System 51
4.3. Aufteilung des Systems 53
4.4. Struktur des Verteilten Systems 55
4.4.1. Methoden und Funktionsaufrufe 55
4.4.2. Data Driven 55
4.4.3. Fazit 59
4.5. Schnittstellen 59
4.5.1. Schnittstellen der Anwendungsfunktionalität 60
4.5.2. Schnittstellen der Filiale 64
4.5.3. Schnittstellen der Infrastruktur 66
4.6. Komponenten 67
4.6.1. Kommunikation 67
4.6.2. Applikation 68
4.6.3. Persistenzserver 71

4.6.4. Benutzerdatenbank 71
4.6.5. Eingabegeräte 71
4.7. Entscheidungen 72
4.8. Zusammenfassung 72

5. Realisierung **73**
5.1. Intention und Umfang 73
5.2. Laboraufbau 73
5.2.1. Serverrechner 75
5.2.2. Projektorrechner 75
5.2.3. Clientrechner 75
5.2.4. Netzinfrastruktur 75
5.3. Programmiersprache 75
5.4. Implementierung der Middleware 76
5.4.1. Blackboardserver 76
5.4.2. Filiale 76
5.4.3. Persistenzserver 77
5.5. Anwendung 77
5.5.1. Fernsteuerung 77
5.5.2. Polylux 82
5.6. Fazit 83

6. Fazit und Ausblick **85**

Literaturverzeichnis **88**

A. Anwendungsfalltabellen **94**

Glossar **103**

Abbildungsverzeichnis

2.1. Zusammenarbeit der Forschungsfelder 20
2.2. Zusammenhang zwischen Arbeitsabläufen und Ideenerzeugung. (Prante u. a. 2002) 25

3.1. Ein BlueBoard in einem Pausenraum 35
3.2. Der persönliche Bereich eines BlueBoards. (Russell u. a. 2002a) 36
3.3. Schema des BlueBoardsystems 36
3.4. Screenshot von SubEthaEdit 38
3.5. Roomware – Second Generation 41
3.6. Aufbau des iRooms 42
3.7. Foto des iRooms aus Stanford 43
3.8. Displaykontrollanwendung im iRoom 44

4.1. Ansicht des System 48
4.2. Sichten auf ein Objekt im System 49
4.3. Zwei Kommunikationsrichtungen 50
4.4. Aufteilung einer Client–Serveranwendungs 50
4.5. Zentraler Terminalserver 51
4.6. Verteilte Architektur 52
4.7. Systemschnittstellen 53
4.8. Verschiedene Anwendungen und Kollaboration 56
4.9. Blackbooardsequenzdiagramm 57
4.10. Architektur eines Blackboards 58
4.11. Schnittstellen einer Applikation 59
4.12. Schnittstellen der Anwendungskomponenten 61

5.1. Aufbau der Laborumgebung 74
5.2. Konfiguration für Fernsteuerung 78
5.3. Der Aufbau des virtuellen Arbeitsraumes 79
5.4. Die Konfiguration des virtuellen Arbeitsraumes, in der Laborumgebung 80
5.5. Konfiguration für die Präsentation 82

Tabellenverzeichnis

2.1. Tool 1 - 3 aus Abb. 2.4.5.1 25

A.1. Anwendungsfall: Systemadministration 95
A.2. Anwendungsfall: Benutzer anlegen 96
A.3. Anwendungsfall: Benutzeraccount löschen 96
A.4. Anwendungsfall: Dienste nutzen 97
A.5. Anwendungsfall: Abrufen von Objekten 98
A.6. Anwendungsfall: Dokument einfügen 99
A.7. Anwendungsfall: Objekt bearbeiten 100
A.8. Anwendungsfall: Objekt anzeigen 100
A.9. Anwendungsfall: Objektverwaltung 101
A.10.Anwendungsfall: Objekt löschen 101
A.11.Anwendungsfall: Objekte einem Projekt zuordnen 101
A.12.Anwendungsfall: Metainformationen editieren 102

1. Einleitung

1.1. Motivation

In Konferenzräumen haben in letzter Zeit verstärkt neue Technologien Einzug gehalten, die die klassischen Hilfsmittel zum Vorbild haben. So sind Videoprojektoren, Netzwerkanschlüsse und verschiedene persönliche Geräte inzwischen allgegenwärtig.

Videoprojektoren erleichtern Präsentationen, da diese nicht mehr auf Folien gedruckt werden müssen. Gleichzeitig ist der Videoprojektor meistens aber nur für einen Teilnehmer zugänglich. Wenn es darum geht schnell zwischen Dokumenten verschiedener Teilnehmer zu wechseln, ist der Nutzen für den einzelnen gering, da für den Austausch von Informationen zwischen den Rechnern umständliche Einstellungen vorzunehmen sind. Durch diese Einschränkung sind Notebooks während des Meetings oft nur für persönliche Notizen nutzbar.

Neue Inhalte werden im Konferenzraum weiterhin mit den klassischen Werkzeugen erstellt, die eine direkte und intuitive Zusammenarbeit ermöglichen. Grund hierfür: Auf Tafeln, Whiteboards und Flipcharts kann von mehreren Teilnehmern gleichzeitig geschrieben werden.

Das verbindende Element der vorgestellten Arbeitsweisen ist der gleichzeitige Zugriff auf ein gemeinsames Medium. Dies gilt für die Aufnahme, genauso wie für das Erstellen von Inhalten.

Die eingesetzten elektronischen Werkzeuge erleichtern die Einzelarbeit und Präsentation. Dagegen ermöglichen die klassischen Werkzeuge das gleichzeitige Schreiben, können jedoch nur von kleineren Gruppen genutzt werden.

> Computers became primary objects of our attention resulting in an area called “human- computer interaction.” Today, however, we must ask: Are we actually interested in interacting with computers? Isn’t our goal rather to interact with information, to communicate and to collaborate with people?
> (Streitz und Nixon 2005)

Seit einigen Jahren tauchen neue Schlagworte in der Literatur auf. Diese beschäftigen sich mit dem Verhältnis zwischen Mensch und Rechner. Sie wollen die Rolle verändern, die der Rechner derzeit einnimmt. Der Computer sollte nicht mehr die dominierende Komponente bei der Interaktion zwischen Mensch und Information sein. Terry Winograd hat in seinem Buch (Winograd und Flores 1987) dargelegt, daß der momentane Umgang mit Rechnern davon geprägt, ist sie in den Vordergrund zu stellen, während die inhaltlichen Tätigkeiten nur an zweiter Stelle stehen. Er fordert ein Umdenken im Softwaredesign, daß Software als Werkzeug begreift. Wie jedes gute Werkzeug sollte sie aus den Gedanken des Benutzers verschwinden, damit er sich ausschließlich seiner Aufgabe widmen kann.

> The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.
> (Weiser 1991)

Einige Jahre nach Terry Winograd hat Mark Weiser in seinem Artikel eine Vision entwickelt: Der Trend zu immer leistungsfähigeren Rechnern in immer kleinerer Formaten, die immer mehr Aufgaben übernehmen, sollte langsam enden. Eine Alternative wären viele Rechner für jeden Benutzer, wobei jeder auf eine Aufgabe spezialisiert ist. Sie sollten ihre Funktionen unauffällig erfüllen, ohne nach Aufmerksamkeit zu verlangen.

Es gibt seit langem Bestrebungen, die Zusammenarbeit innerhalb von Gruppen durch Software, der sogenannten Groupware, zu verbessern. Groupware ein Kunstwort aus "**group** processes" und der "soft**ware**", um sie zu unterstützen (Johnson-Lenz und Johnson-Lenz 1994). Sie wird beschrieben als Software, die Gruppen von Menschen bei einer gemeinsamen Aufgabe unterstützt und eine Schnittstelle zu einer geteilten Umgebung bietet (Ellis u. a. 1991). Das Entwickeln von Groupware ist eine Aufgabe, mit der sich das Feld der "Computer Supported Collaborative Work" befasst.

Diese drei Strömungen können alle in einem Konferenzraum verwirklicht werden. Dafür soll eine Groupwareentwickelt werden, die den Ideen Winograds folgt und Geräte nutzt, die in die Umgebung des Konferenzraumes eingebettet sind. Durch die gemeinsame Nutzung der elektronischen Medien und die Integration der klassischen Werkzeuge, ergibt sich ein großes Potenzial für Verbesserungen.

Dieser Aufgabe haben sich mehrere Teams gewidmet. Sie haben Hardwareprototypen mit einer gemeinsamen Software verbunden. Zwei dieser Projekte sind "Roomware"[1], das vom Fraunhofer Institut Darmstadt entwickelt wurde, und das "Interactive Workspaces"-Projekt[2] der Universität Stanford. Beide Arbeitsgruppen haben sich allerdings auf die Implementierung der Groupwarefunktionalität ((Johanson u. a. 2001, 12) und (Tandler 2004, 214)) und

[1] Siehe Abschnitt 3.5.1 auf Seite 40
[2] Siehe Abschnitt 3.5.2 auf Seite 42

den Entwurf von Geräteprototypen ((Guimbretière u. a. 2001) und (Tandler u. a. 2002, 38-40)) konzentriert.

Während solche Projekte bisher Forschungslaboren vorbehalten waren, ist der Stand der Technik inzwischen soweit fortgeschritten, daß auch regulär erhältliche Geräte genutzt werden können, von großen Displays oder Projektoren bis zu PDAs mit WLAN. Auch die Sicherheit soll nicht länger ausgeklammert bleiben und einen Platz im System finden.

Mit Hilfe von vorhandenen Techniken des Ubiquitous Computing und neuer Benutzerschnittstellen kann man nun ein System entwerfen, das die Vorteile der Rechner nutzt, ohne auf die Vorteile der gewohnten Hilfsmittel zu verzichten. Standen bis jetzt Computer für sich alleine, sind sie nun Teil eines Ganzen. Waren Tafel und Whiteboard noch vollkommen getrennt von elektronischen Systemen, werden sie jetzt in das neue System integriert.

1.2. Szenario

Das vorliegende Szenario beschreibt einen Konferenzraum und damit einen Ort, der für verschiedene Aufgaben genutzt wird: Brainstorming, Präsentation, Diskussion und die Strukturierung von Ideen charakterisieren eine möglichst zielgerichtete Arbeitsatmosphäre. Im Folgenden wird ein fiktives Meeting beschrieben und die Vorteile und Möglichkeiten aufgezeigt, die sich aus der Verschmelzung der Technologien ergeben.

1.2.1. Ablauf eines fiktiven Meetings

Alice, Bob, Charlie und Dave treffen sich zu einem Arbeitsgespräch. Alice und Bob haben vorbereitete Dokumente dabei.

Im Konferenzraum steht ein Videoprojektor, der an einen Server des Systems angeschlossen ist. Der Konferenzraum bietet einfache Möglichkeiten des Netzwerkzugangs.

Die Computer der Teilnehmer erkennen das Netz des Konferenzraums und bauen selbstständig eine Verbindung auf. Die laufende Clientsoftware erkennt die vorhandenen Dienste des Raumes und meldet ihre Anwesenheit. Der Server gibt einen Überblick aller momentan angemeldeten Teilnehmer dieses Meetings.

Alice beginnt, lässt ihr Dokument anzeigen und erklärt dessen Inhalt. Während sie spricht, haben Bob und Charlie weitere Ideen, die sie als Anmerkungen an Alices Dokument anhängen und die neben dem eigentlichen Inhalt angezeigt werden. Alice hebt einige Elemente hervor und verändert andere. Diese Aktivitäten und die dadurch veränderten Daten werden vom Server in einem Archiv gesichert, so daß sie später wieder genutzt werden können.

Später wiederholt sich der Vorgang bei Bobs Präsentation, die er von seinem persönlichen Notebook starten kann. Dave ist während dieser Diskussion eine Idee gekommen, die er mit einem Dokument vorstellen möchte. Er hat es bereits während des Meetings aus seinen Notizen erstellt und wählt das öffentliche Display zur Darstellung. Innerhalb kürzester Zeit wird es vom Server dargestellt und kann diskutiert werden.

Das projizierte Bild enthält nun nicht nur die Daten der Teilnehmer, sondern auch Informationen über den bisherigen Verlauf des Meetings. In einem gesonderten Bereich ist das aktuelle Dokument mit den Anmerkungen und Markierungen der Benutzer zu sehen. Durch die dargestellte Historie kann ein neuer Teilnehmer einen groben Eindruck über den Verlauf des Treffens und den aktuellen Diskussionsstand bekommen.

Sollte er mit den gezeigten Dokumenten nicht vertraut sein, kann er diese, zusammen mit den Anmerkungen, auf einem eigenen Display anzeigen lassen.

Nach Ende des Meetings, durch Abmelden aller Teilnehmer, wird auf dem Server eine Liste erstellt, die alle erzeugten Dokumentenversionen verlinkt. Diese Liste wird den Teilnehmern zugestellt.

Das Beispielmeeting zeigt, wie verschiedene Tätigkeiten durch Vernetzung erleichtert wurden. So konnten neue Dokumente ohne Umstände vorgestellt und von den Teilnehmern kommentiert werden. Die erzielten Ergebnisse waren sofort zugänglich, ohne daß die Teilnehmer dies veranlassen mussten. Bei späteren Meetings können diese Dokumente nun weiter genutzt werden.

1.2.2. Gezeigte Anwendungen

Im vorherigen Abschnitt wurden Anwendungen aufgezeigt, welche die Aufgaben im Konferenzraum erleichtern sollen. Vorhandene Dokumente konnten einfach vorgestellt werden, unabhängig davon, ob sie vorbereitet waren oder "on the fly" erstellt wurden. Dies wurde durch die Vernetzung der darstellenden Geräte mit den Speicherorten der Dokumente und einer Steuerung für die darstellenden Geräte möglich. Eine solche Vernetzung von Inhalten mit einer einfachen Möglichkeit der Präsentation wird auch von BlueBoard demonstriert.[3]

Im vorgestellten Beispiel konnten Anmerkungen einfach erstellt und an dargestellte Dokumente angehängt werden. Es wurde vermieden, daß die Teilnehmer zunächst mit ihren Kommentaren auf das Ende eines Redebeitrags warten mussten. In (Prante u. a. 2002) wurde beobachtet, daß dies einen großen Unterschied in der Produktivität ausmachen kann. Alle erstellten Inhalte wurden archiviert, so daß die Teilnehmer sich nicht mit der Speicherung

[3] Bei diesem Projekt von IBM wird ein vernetzter Rechner mit einem Touchscreen aufgestellt. Nach der Anmeldung mit einer RFID-Plakette ist ein persönlicher Informationsbereich zugänglich. (Russell u. a. 2005)

ihrer Dokumente beschäftigen mussten. Sie konnten sich auf die inhaltlichen Aufgaben konzentrieren.[4]

1.3. Ziel der Arbeit

Das Ziel der vorliegenden Arbeit ist, die Anforderungen des Szenarios zu analysieren und eine Architektur für die Realisierung des Szenarios zu entwerfen. Die Architektur soll modular aufgebaut sein, um spätere Erweiterungen zu ermöglichen. Als Beispielszenario dient ein Softwareentwicklungsteam, daß bei der Gruppenarbeit in Konferenzräumen unterstützt werden soll. Eine genauere Ausarbeitung von Szenarien und ihren Anforderungen finden sich in (Neumann 2006) und (Burfeindt 2006). Das entworfene System soll daraufhin als Prototyp realisiert werden. Hierbei übernehmen normale Desktop-PCs die Rollen der Endgeräte. Die Nutzung der gebotenen Dienste sollte aber auch von einem PDA aus möglich sein.

1.4. Gliederung der Arbeit

In Kapitel 2 wird das Szenario beschrieben und die daraus folgenden Anforderungen erarbeitet. Darauf aufbauend werden in Kapitel 3 auf dem Markt erhältliche Produkte und ihre Eigenschaften untersucht. Sie werden mit den Anforderungen verglichen und eventuell für die Umsetzung Vision verwendet. Danach wird in Kapitel 4 ein Design entworfen, daß sich aus den Anforderungen ableitet. In Kapitel 5 wird die prototypische Realisierung des Frameworks und zweier Anwendungen beschrieben. Gegen Ende wird in Kapitel 6 ein Überblick über die Arbeit, erreichte Ziele und zukünftige Themengebiete gegeben.

[4] Wobei hier nur Aktionen archiviert werden, die über die Anwendung ausgeführt werden. Tivoli (ein weiteres CSCW Tool) sieht eine komplette Audioprotokollierung vor, durch die wichtige Gesprächsideen nicht verloren gehen. Es muss gewährleistet sein, daß die gespeicherten Daten nicht den Kreis der Teilnehmer verlassen. Weitere Probleme entstanden aus der Verbindung von offensiven Bemerkungen mit der dauerhaften Speicherung. (Scholtz u. a. 2003)

2. Analyse

2.1. Rahmenbedingungen und Eigenschaften der Zielgruppe

Groupware unterstützt und erleichtert die Zusammenarbeit einer Gruppe mit einem gemeinsamen Ziel. Die eingesetzten Mittel unterscheiden sich hierbei abhängig von der Arbeit der Gruppe und ihrer Zusammensetzung, sowie den Rahmenbedingungen, unter denen sie arbeiten muss. Ein Produkt für die Unterstützung eines weltweit verteilten Teams stellt andere Werkzeuge zur Verfügung, als eines für die Unterstützung eines lokalen Teams. In den folgenden Abschnitten werden die Rahmenbedingungen der Arbeit sowie die daraus folgenden Eigenschaften unter drei Aspekten beschrieben. Hierzu zählt die Art der Arbeit und der Gruppe. Sowie die Art der Unterstützung durch das System und die Gesprächsführung. Der dritte Aspekt sind Annahmen über die technischen Vorraussetzungen.

2.1.1. Zeit, Ort und Art der Gruppe

Dieser Abschnitt stellt die Art der Benutzergruppe, ihre Ziele und ihre Arbeitsbedingungen vor.

Das System wird von Teams genutzt, die ein gemeinsames Ziel verfolgen. Zum Erreichen dieses Zieles können mehrere Monate nötig sein, über die die Ergebnisse ihre Gültigkeit behalten. Wurden einmal verbindliche Vereinbarungen getroffen, muss diese Fassung vor weiteren Änderungen geschützt werden. Dies ist ein Prozess, der kontinuierlich Ergebnisse schafft und auf ihnen aufbaut.

Ein beispielhaftes Umfeld wäre eine Firma mit etwa 50 Mitarbeitern Die Mitarbeiter werden projektbezogen zu Teams mit je fünf bis zehn Personen zusammengefasst. Einzelne Projekte dauern drei Monate oder länger. Die Teams lösen sich teilweise ab. Die Teams treffen untereinander und die Firma mit Geschäftspartnern Vereinbarungen. Beide Formen von Vereinbarungen müssen dauerhaft gespeichert werden.

Die festen Teams von überschaubarer Größe verfolgen durch kontinuierliche Arbeit über einen längeren Zeitraum ein gemeinsames Ziel. Im Laufe der Zusammenarbeit treffen sich

die Mitglieder des Teams immer wieder an einem gemeinsamen Ort, um bestimmte Themen zu bearbeiten. Diese Treffen haben eine überschaubare Teilnehmerzahl. Im Vorfeld werden einige Dokumente vorbereitet, andere während eines Treffens gemeinsam erstellt.

Während spontane Treffen nicht ausgeschlossen sind, haben die Ergebnisse doch zeitlich weiterreichende Konsequenzen und müssen entsprechend gespeichert werden.

Die Anforderungen bestimmter Gruppen an ein solches System variieren je nach Fachgebiet und werden von (Neumann 2006) am Beispiel eines Softwareentwicklungsprozesses und (Burfeindt 2006) anhand eines Stadtplanungsprozess weiter ausgeführt. Die später aufgeführten fachlichen Anforderungen sind aus diesen Arbeiten hergeleitet, und bilden die gemeinsame Basis für die Bearbeitung Prozesse.

2.1.2. Steuerung und Usability

Es gibt verschiedene Möglichkeiten die Zusammenarbeit einer Projektgruppe zu modellieren. Aus der Softwareentwicklung sind mehrere Ansätze bekannt die einen starren Entwicklungsprozess vorgeben z. B. das Wasserfallmodell. Modernere Ansätze zur Softwareentwicklung stimmen jedoch darin überein, daß sie keinen festen Ablauf mehr fordern. Stattdessen wird auf mehrere Iterationen gesetzt, in denen das Endprodukt zu zunehmender Reife gelangt, während die Entwicklung sich an neue Erkenntnisse über die Ziele des Projektes anpasst. Ein Beispiel für ein solches Vorgehensmodell ist Extreme Programming, welches die Idee der Iterationen aufnimmt und verstärkt. Hier kommen sehr kurze Iterationen vor die teilweise nur eine Woche dauern (Beck und Andres 2004). Entsprechend soll das System keine festen Workflows vorgeben sondern jederzeit in allen Formen nutzbar sein. Die einzige Struktur der Kommunikationssteuerung wird von den Benutzern selbst vorgegeben. Hierdurch ist es jederzeit möglich, den Raum zu betreten und frühere Dokumente weiterzuentwickeln.

Trotz des Einsatzes in langfristigen Szenarios, soll die Benutzung ohne umfangreiches Training möglich sein. Die Funktionen des Systems sollen vor der Erfüllung der Aufgaben in den Hintergrund treten.

Die Verwaltung der Teilnehmer kann von Benutzern mit entsprechenden Rechten oder ihrer Delegierten übernommen werden. Diese können die Aufnahme eines Teilnehmers in eine Gruppe als Mitglied oder Gast veranlassen. Solche Entscheidungen gelten dauerhaft, bis sie explizit aufgehoben werden.

2.1.3. Systemannahmen

Beim Entwurf dieses Systems werden einige Annahmen getroffen, die Auswirkungen auf die Anforderungen und das spätere Design haben werden. Es gibt bei CSCW-Lösungen das Prinzip des "What you see is what I see". Dies ist bei verteilten Teams wichtig, um eine Basis für Gespräche zu erhalten. Da in diesem System alle Mitarbeiter Zugriff auf dieselben Ausgabegeräte haben ist es in diesem Fall nicht notwendig, ja sogar nicht umsetzbar, da ja PDAs und Wanddisplays eingesetzt werden sollen. Es wird versucht einen einheitlichen Datenzustand zu wahren, so daß jeder Benutzer die gleichen Informationen sieht.

Ein weiterer Aspekt bei Groupwaresystemen ist die Fragestellung der Floor–Control, d. h. die Kontrolle über die Gesprächsführung. Bei verteilten "Computer Supported Collaborative Work"-Produkten[1] gibt es hierfür Protokolle, die die Kontrollübergabe regeln. Bei unserer Lösung wird davon ausgegangen, daß die Teilnehmer sich untereinander absprechen. Dadurch kann auf technische Maßnahmen verzichtet werden, so daß die Lösung flexibler wird. Die Erfahrungen in (Russell u. a. 2002a) zeigen, daß solche Kontrollmaßnahmen in der Regel, nicht nachgefragt werden.

Bezüglich der Sicherheitsmaßnahmen wird von einer gutmütigen Umgebung ausgegangen. Die beteiligten Softwarekomponenten halten sich an die vereinbarten Protokolle. Hierfür ist es notwendig, die Teilnahme fremder Geräte am Netzwerk zu unterbinden. An einigen Geräten können mehrere Benutzer gleichzeitig arbeiten, hier wird die Verantwortung für die ausgeführten Aktionen auf die angemeldeten Benutzer übertragen. Die Benutzer kontrollieren gegenseitig, daß niemand unerwünschte Änderungen unter falschem Namen veranlasst.

Das fertige System sollte Informationen über die Umgebung nutzen, dies wird auch als Awareness bezeichnet. Dies kann die Notwendigkeit expliziter Anmeldevorgänge reduzieren und den Zugriff auf einen persönlichen Datenraum erleichtern. Die Entscheidungen beschränken sich auf einfache Fälle. So soll vermieden werden, den Benutzer durch falsche Entscheidungen zu behindern.

2.2. Beschreibung des Status Quo

Ein Konferenzraum ist ein Ort, in dem verschiedene Gruppen zusammenkommen, um gemeinsam eine Aufgabe zu erledigen. So vielfältig wie die Gruppen sind auch die Aufgaben. Trotz aller Unterschiede haben sich eine Reihe von Hilfsmitteln etabliert, die in vielen Konferenzräumen zu finden sind. Diese Hilfsmittel unterstützen die gemeinsame Arbeit der Gruppen, ohne Annahmen über ihre Art zu treffen. Sie konzentrieren sich auf die Gemeinsamkeiten der Gruppenarbeit, das Mitteilen, Diskutieren und Erarbeiten von Wissen und Ideen.

[1] CSCW

Unterschiede im Fachbereich äußern sich durch Verwendung spezialisierter Werkzeuge und die behandelten Inhalte.

Für einige dieser Tätigkeiten, wie das Diskutieren und Erarbeiten von Wissen und Ideen, haben sich Moderationstechniken entwickelt, die die Erhöhung der Effizienz und Effektivität zum Ziel haben. Solche Techniken bedienen sich der klassischen Hilfsmittel, wie z. B. Whiteboard oder Karteikarten.

Die Arbeit im Konferenzraum ist meistens lokal und gleichzeitig. Es sind selten entfernte Mitarbeiter beteiligt. Dies reflektiert sich auch in den installierten Hilfsmitteln. Sie bieten selten die Möglichkeit, Ergebnisse dauerhaft zu speichern und später weiter zu verwenden. Die Werkzeuge unterscheiden sich in Reichweite und der möglichen Interaktion. So können Whiteboard und Tafel von mehreren Benutzern gleichberechtigt verwendet werden, jedoch begrenzt auf eine kleinere Gruppe, auf Grund der Größe der Tafel. Die Bilder von Overhead– und Videoprojektor können für ein großes Publikum vergrößert werden. Sie können allerdings nur von wenigen Benutzern parallel verwendet werden.

Zusätzlich werden meistens viele technische Geräte in Besprechungen mitgenommen, die noch nicht zur Zusammenarbeit genutzt werden. Stattdessen dienen sie als persönliche Werkzeuge, da die Integration in die vorhandene Infrastruktur zu umständlich ist. Als Beispiel kann die Nutzung des Videoprojektors gesehen werden. Hierbei muss die Präsentation auf den im Idealfall vernetzten Präsentationsrechner kopiert werden und von diesem lokal gesteuert werden. Alternativ kann ein Notebook an den Grafikeingang des Videoprojektors angeschlossen werden.

Verschiedene Techniken, die im Status Quo beschrieben wurden, benötigen Verbrauchsmittel, die ersetzt werden müssen. Die ermittelten Ergebnisse können nicht auf mehreren Werkzeugen bearbeitet oder direkt festgehalten werden. Sollte eine Archivierung gewünscht sein, muss die Tafel oder das Whiteboard abfotografiert bzw. die Folien kopiert werden. Dies muss bei Whiteboards, Tafeln und arrangierten Karteikarten direkt im Anschluss geschehen, da die nächste Nutzung sie löschen würde. Eine direkte Zusammenarbeit ist nur zwischen Personen möglich, die direkt vor den Whiteboards stehen. Dadurch ist die maximale Zahl an Mitarbeitern stark begrenzt.

Für die einzelnen Techniken gibt es Alternativen, die Teilaspekte verbessern. Allerdings existiert bisher kein umfassendes Konzept, das diese Einzelteile zu einem Ganzen verbindet. So gibt es z. B. Smartboards, die alle erfolgten Eingaben mitzeichnen und später elektronisch zur Verfügung stellen. Sie sind aber weder untereinander, noch mit den mitgebrachten Geräten der Teilnehmer vernetzbar.

2.3. Szenario (Soll-Zustand)

Auch in einem Konferenzraum der Zukunft sollten die Werkzeuge so einfach wie die heute vorhandenen Hilfsmittel zu bedienen sein, obwohl sie weit mehr Möglichkeiten bieten. So können Inhalte, die auf einem Gerät erstellt werden, auch auf weiteren Geräten angezeigt und parallel bearbeitet werden. Der Stand nach jeder Eingabe wird, von den Benutzern unbemerkt, aufgezeichnet und kann bei Bedarf wieder hervorgeholt werden. Hierdurch ist es möglich, bei einem Folgetreffen nahtlos mit einem vorherigen Stand, oder auch mit zwischenzeitlich aktualisierten Versionen weiterzuarbeiten. In dieser Form fallen große Datenmengen, sie ist also nicht langfristig durchzuhalten. Um dieses Problem zu lösen, können nach einer festgelegten Menge von Änderungen Zusammenfassungen angelegt und die Rohdaten gelöscht werden. Bei diesen Aktionen ist darauf zu achten, daß die Zuordnung von Benutzer zu Änderung erhalten bleibt.

Sollten weitere Treffen in anderen Räumen stattfinden, so kann auch dort auf die aktuellen Daten zugegriffen werden. Dokumente können jederzeit nach Autor oder Änderungsdatum gesucht werden. Insgesamt soll das System die Effektivität und Effizienz von Gruppenarbeit steigern. Dies geschieht durch Einbinden der ansonsten passiven Teilnehmer, da sie nun auch parallel mitarbeiten können, anstatt ohne einen Gesprächsbeitrag abwarten zu müssen. Weiterhin wird der Aufwand für die Präsentation neuer Inhalte verringert. Und schließlich werden die Ergebnisse des Gesprächs bereits während des Gesprächsverlaufs gemeinsam erarbeitet.

Die Konzentration liegt auf allgemeinen Funktionen, die in vielen Fachbereichen gebraucht werden. Weitere Aufgabe ist die Schaffung einer Infrastruktur, die diese erweiterten Möglichkeiten den Teilnehmern und fachspezifischen Anwendungen zur Verfügung stellt.

2.3.1. Einfaches Mitteilen von Informationen

Inhalte können den anderen Teilnehmern einfach und schnell mitgeteilt werden. Wichtig ist: Auch in einer Fülle von Informationen müssen einzelne Inhalte schnell und zuverlässig aufzufinden sein. Zusätzlich müssen sie einfach auf allgemein sichtbaren Flächen dargestellt werden können. In (Knop 2004) wurde bereits das Thema der Informationsverwaltung für persönliche Systeme untersucht. Diese Erkenntnisse können auf den Informationszugriff in einem Mehrbenutzersystem übertragen werden.

2.3.2. Gemeinsames Bearbeiten

Die Teilnehmer eines Treffens sollen auf dieselben Dokumente zugreifen und diese gleichzeitig bearbeiten können. Die Dokumente befinden sich in einem virtuellen Raum, auf den jeder Teilnehmer Zugriff hat. Dadurch spielen bei der Zusammenarbeit die Grenzen des eigenen Gerätes keine Rolle mehr. Die einzelnen Benutzer haben hierbei Zugriff auf die aktuelle Version, welche bei Änderungen anderer Benutzer aktualisiert wird.

2.3.3. Einfaches Festhalten von Ideen

Ideen, die während eines Treffens besprochen werden, müssen nicht explizit gespeichert werden, sondern das System verwaltet die Änderungen. Das Erstellen von Inhalt soll leicht und schnell möglich sein. Gleichzeitig müssen Veränderungen, zusammen mit den impliziten Metadaten wie Autor, Zeit, und Ort, ohne Interaktion gespeichert werden. Durch diese und zusätzliche Metadaten sind die Dokumente problemlos auffindbar.

2.3.4. Weitere Forderungen

Zusätzlich soll das System des Konferenzraums die Möglichkeiten nutzen, die eine Vernetzung und die computergestützte Erstellung und Speicherung der Inhalte mit sich bringt.

Um die Nutzung möglichst intuitiv zu gestalten, sollen sich die neuen Funktionen in die vorhandene Wahrnehmung des Benutzers eingliedern. Das heißt, daß elektronische Whiteboards und andere Peripheriegeräte, sich genau wie ihre ältere Entsprechung nutzen lassen. Auf Notebooks gliedern sich die neuen Funktionen in die Betriebssystemoberfläche ein. Um diese Ziele zu erreichen, sollen, soweit möglich, vorhandene Produkte genutzt werden.

Zusätzlich kommen noch die Sicherheitsanforderungen eines Mehrbenutzersystems hinzu. So sollen Dokumente nur von berechtigten Benutzern betrachtet bzw. verändert werden können, weshalb das System eine Authentifizierung und Autorisierung der Benutzer unterstützen muss. Hierfür soll ein Konferenzraum als eine Einheit betrachtet werden, bei dessen Betreten man sich anmelden muss, genau wie an den einzelnen Eingabegeräten.

Sollte es sich während eines Treffens ergeben, daß Unterpunkte besser von kleineren Teilgruppen bearbeitet werden, kann sich das Team aufteilen. Hierbei behalten die Teilgruppen den Zugriff auf die nötigen Files. Sobald der Kontakt zum System abbricht, wird ein Branch des Dokumentes angelegt, der später mit der zentral gespeicherten Version abgeglichen werden sollte. Die neuen Funktionen werden an die Größe und Leistungsfähigkeit der verschiedenen Endgeräte angepasst, so daß die Endgeräte im System verwendet werden können. Sobald ein Benutzer sich am Raum angemeldet hat, können seine mitgebrachten

Geräte als Komponenten des Systems fungieren und ihm die erweiterten Möglichkeiten anbieten.

2.4. Technische Anforderungen

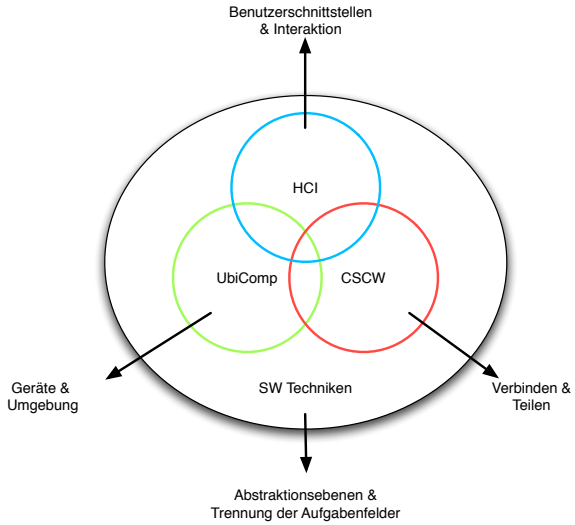

Abbildung 2.1.: Zusammenarbeit der Forschungsfelder aus (Tandler 2003)

Wie in den vorherigen Abschnitten dargestellt, soll der Raum für unterschiedliche Anwendungen nutzbar sein. Zwei solcher Anforderungsprofile werden exemplarisch in (Neumann 2006) und (Burfeindt 2006) entwickelt. Die unterschiedlichen Szenarien aus den Bereichen Softwareentwicklung und Stadtplanung haben eine Schnittmenge von Anforderungen. Diese sollen durch das System erfüllt werden.

Die folgenden Abschnitte beschäftigen sich mit den gemeinsamen Anforderungen und stellen dar, welche Möglichkeiten den darauf aufbauenden Entwicklungen geboten werden. Die Anforderungen an solche Systeme kommen aus verschiedenen Bereichen der Informatik, die hier einen Berührungspunkt haben. Diese Verbindungen werden in Abbildung 2.1 dargestellt und beinhalten die Themengebiete Mensch–Computer–Interaktion (MCI), Computer–Supported–Collaborative–Work (CSCW), Ubiquitous Computing (UbiComp) sowie die Softwaretechnik. Aus dem CSCW–Bereich kommen die Anforderungen an die Bearbeitungstechniken. Ubiquitous Computing formuliert Anforderungen an die spontane Integration mobiler Geräte. Aus der Softwaretechnik kommen Anforderungen an die Architektur und Implementierung.

2.4.1. Benutzer – Geräte Konfigurationen

Bei den Treffen kommen mehrere Benutzer zusammen, die ihre privaten Geräte mitbringen. Diese sollen ins System integriert werden und vorhandene sowie gegenseitig zur Verfügung gestellte Dienste nutzen. Im Raum selbst steht eine Reihe von öffentlichen Geräten zur Benutzung bereit. Es ist somit ein Mehrbenutzersystem, das aus einer heterogenen Menge von öffentlichen und privaten Geräten besteht. Außerdem besteht die Möglichkeit, daß ein Benutzer mehrere Geräte gleichzeitig verwendet.

Die daraus folgenden Konstellationen müssen ermöglicht werden, ohne weitere Anforderungen, wie z. B. an die Sicherheit, zu verletzen. Eine genauere Ausarbeitung der Anforderungen an die Benutzerverwaltung und das Rechtemodell erfolgt in (Mund 2006). In den folgenden Absätzen werden nur die möglichen Benutzer–Geräte–Konfigurationen aufgezählt und ihr Auftreten in diesem System begründet. Die Konstellationen müssen nicht in der Infrastruktur ermöglicht werden, auch die Benutzerschnittstelle muss sie angemessen repräsentieren.

2.4.1.1. Ein Benutzer — Mehrere Geräte

Ein Benutzer wird häufig an mehreren Geräten arbeiten oder zumindest angemeldet sein. Seine Eingaben an diesen Geräten sollen ihm zugeordnet werden, um später den Urheber einer Änderung zu erkennen.

Ein Beispiel wäre ein Vortrag, bei dem der Benutzer ein Kontrolldisplay mit Notizen nutzt sowie große, interaktive, öffentliche Displays[2], über die seine Präsentation läuft. Alternativ wäre es denkbar, daß ein PDA Nachrichten für einen Benutzer sammelt, während der Benutzer an einem fest installierten Touchscreen der Präsentation folgt.

2.4.1.2. Viele Benutzer — Mehrere Geräte

Dasselbe Dokument wird von mehreren Benutzern an verschiedenen Geräten betrachtet und zusammen editiert. Hierbei sollten die einzelnen Endgeräte nicht darauf angewiesen sein, die anderen Geräte zu kennen, um das parallele Bearbeiten zu ermöglichen.

Ein Beispiel ist ein größeres Objekt z. B. ein Projektplan, bei dem die Benutzer versuchen, einen Konsens zu finden und ihre Vorschläge einzuarbeiten oder bei dem die Benutzer gemeinsam einen längeren Text formulieren.

[2] Large Interactive Public Displays (LIPD)

2.4.1.3. Ein Gerät — Mehrere Benutzer

An Geräten mit größerem Bildschirm können mehrere Benutzer arbeiten, entweder am selben oder an verschiedenen Dokumenten. Dies tritt auf, wenn ein Dokument auf einem Smartboard dargestellt wird und eine Gruppe von Benutzern es bearbeitet. Eine Schwierigkeit dabei ist, die Änderungen einem einzelnen Benutzer zuzuweisen, um den Urheber einer Änderung speichern und unberechtigte Zugriffe verhindern zu können. Dies kann durch die Verwendung spezieller, identifizierbarer Eingabegeräte erfolgen oder durch die manuelle Bekanntgabe des aktuellen Benutzers, z. B. mit einer Fläche des Displays, die vor Eingaben gedrückt werden muss. Ein solches Problem kann man umgehen, indem man das Konzept einer gemeinsamen Autorschaft einführt.

2.4.2. Wissen um die Raumumgebung

Durch weitere Sensoren oder Eingabegeräte kann die Menge an verfügbaren Informationen über die Umgebung zunehmen. Das System muss flexibel genug sein, die Sensoren und ihre Daten zu integrieren. Sofern verfügbar, sollten das System und weitere Anwendungen diese zusätzlichen Informationen nutzen.

2.4.2.1. Raumzugangskontrolle

In diesem System gibt es Geräte die öffentlich einsehbar sind bzw. keine Anmeldevorrichtung haben. Daher muss es eine raumweite Authentifizierung geben, die Benutzer vor Betreten des Raums durchführen. Dabei melden sich die Benutzer beispielsweise an einer spezialisierten Anmeldestation an, die sich vor dem Eingang des Raumes befindet. Die Form des Anmeldevorgangs ist hierbei nicht explizit festgelegt, sie muss lediglich eine einwandfreie Identifizierung erlauben. Dadurch kennt das System die Namen der Benutzer und kann mit geeigneten Schritten reagieren.

Zu den geeigneten Schritten, gehören Maßnahmen zur Vermeidung von unberechtigten Objektzugriffen, das Vorbereiten der Arbeitsbereiche für die neu hinzugekommenen Benutzer und das Darstellen ihrer Anwesenheit auf einer aktuellen Benutzerliste des Raumes. So können sie schnell auf vorbereitete Dokumente zugreifen, die für alle Benutzer im Raum lesbar sind.

2.4.2.2. Wissen um Benutzer

Die Position der Benutzer kann verwendet werden, um ihnen an Wanddisplays automatisch Zugriff auf ihren Dokumentenbereich zu geben und sie bei Änderungen automatisch als Autor zu erfassen. Sollte sich ein Benutzer, ohne die nötigen Rechte, dem Display nähern, kann eine Zugriffsverletzung verhindert werden. Wenn zwei Benutzer nebeneinander sitzen, können zusätzliche Möglichkeiten der Zusammenarbeit zur Verfügung gestellt werden.

2.4.3. Objektpersistenz

Wie in Abschnitt 2.1 ausgeführt, unterstützt das System eine Gruppe von Benutzern bei langfristigen Aufgaben. Daraus folgt, daß Dokumente, die in einer Sitzung erzeugt wurden, auch bei nachfolgenden Treffen verfügbar sein müssen, um einen kontinuierlichen Arbeitsfluss zu gewährleisten. Ansonsten müssten sich die Benutzer um die Sicherung und Verwaltung der Dokumente kümmern. Dies gilt sowohl für vorbereitete Dokumente, die im Meeting eingebracht werden sollen, als auch für Inhalte, die während einer Sitzung erarbeitet wurden.

2.4.3.1. Ortstransparenter Zugriff

Nach dem Treffen werden die erstellten Dokumente auch in der täglichen Arbeit genutzt. Um das zu gewährleisten, muss ein ortstransparenter Zugriffs, unter Beachtung des Sicherheitsmodells, möglich sein. Die Inhalte des Systems sollen für die Benutzer überall verfügbar sein, ohne Informationen über den logischen Pfad oder den physischen Aufenthaltsort der Daten zu haben. Sie sollen an ihren privaten Arbeitsplätzen und in den Gemeinschaftsräumen eine gewohnte Struktur vorfinden.

2.4.3.2. Versionskontrolle

Die Gemeinschaftsarbeit kann dazu führen, daß sich Dokumente häufig ändern. Um Vergleiche zu früheren Diskussionsergebnissen ziehen zu können, müssen alte Versionen gespeichert und verfügbar bleiben. Sollte sich herausstellen, daß ein vorhergehendes Ergebnis besser mit den Zielen der Teilnehmer übereinstimmt, können erarbeitete Ergebnisse verworfen und zu einer früheren Version zurückgekehrt werden.

Um Änderungen nachvollziehbar zu machen und um die Auffindbarkeit von Objekten zu erhöhen, werden auch Metadaten zu den einzelnen Dokumenten und ihre Versionen gespeichert. Da diese teilweise editierbar sind, müssen auch die Metadaten Informationen über

ihren Autor enthalten und unter einer Versionskontrolle stehen. So kann bei Bedarf das komplette System auf einen früheren Stand gesetzt werden.

2.4.4. Darstellung

Beim Arbeiten im Raum werden Objekte unterschiedlichen Typs genutzt. Diese werden auf verschiedenen Geräten bearbeitet und verwaltet, die sich schon in ihren Abmessungen stark unterscheiden. Alleine bei der grafischen Darstellung reicht die Spannbreite von 2*1.5 Metern Projektionsfläche, bis hinunter zu 10*20cm beim PDA. Außerdem in Zukunft Geräte ohne Grafikausgabe eingesetzt werden. Damit die Benutzer trotz dieser Unterschiede zusammenarbeiten können, muss die Umgebung die Darstellung an die Gerätemöglichkeiten anpassen. Zudem müssen Informationen über die Lage der Geräte verwendet werden können. So reicht einem PDA–Nutzer manchmal eine Icon des Objekts, z. B. wenn die Inhalte des Datenobjekts gleichzeitig auf einem LIPD erscheinen.

2.4.5. Bearbeiten

Der Raum soll nicht nur das Darstellen von Objekten erlauben, sondern auch ihre Bearbeitung. In diesem Abschnitt werden zwei Themen die hiermit zusammenhängen besprochen. Unterstützte Arbeitsmodi und ein Satz allgemeiner Operationen, die anwendungsunabhängig möglich sein sollen, damit Inhalt einfach erstellt werden kann.

2.4.5.1. Arbeitsmodi

Es gibt verschiedene Möglichkeiten, die parallele Arbeit an einem Dokument zu ermöglichen. Sie gehen alle auf das Verwalten von konkurrierenden Zugriffen auf Objekte zurück. Es gibt hierbei die gegenläufigen Ziele, die Wartezeit zu verkürzen oder Konflikte bei gleichzeitigen Bearbeitungsschritten zu vermeiden.

Das Verringern der Wartezeit bedeutet, daß Benutzer nicht warten müssen, bis ihnen die Kontrolle übergeben wird, sondern jederzeit am Dokument arbeiten können. Konflikte entstehen, wenn zwei Benutzer die gleiche Dokumentenstelle gleichzeitig bearbeiten und widersprüchliche Änderungen erzeugen. Diese Konflikte können nicht automatisch aufgelöst werden, da hierfür ein Verständnis der Inhalte nötig ist.

Die unterschiedlichen Ansätze siedeln sich nun, je nach ihrer Zielsetzung, an verschiedenen Stellen zwischen diesen Polen an.

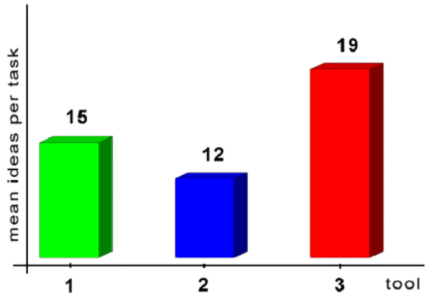

Abbildung 2.2.: Zusammenhang zwischen Arbeitsabläufen und Ideenerzeugung. (Prante u. a. 2002)

Structure Work Mode	Synchronous	Turn-taking
Mind-Mapping	N/A	Tool 1
Whiteboard	Tool 3	Tool 2

Tabelle 2.1.: Tool 1 – 3 aus Abb. 2.4.5.1 aufgeschlüsselt nach Anwendungsgebiet und Arbeitsmodus

In (Prante u. a. 2002) wurde die Effektivität verschiedener Tools zur Ideenfindung untersucht. Dort ließ man Gruppen von Informatikstudenten in einer festen Zeit Lösungen für eine gestellte Aufgabe suchen. Den Gruppen wurden unterschiedliche Werkzeuge zur Verfügung gestellt und die Ergebnisse verglichen. Diese Werkzeuge waren je ein elektronisches Whiteboardsystem mit serieller und paralleler Arbeitsweise, sowie Mindmappingtool mit serieller Arbeitsweise. In der darauf folgenden Untersuchung der Ergebnisse stellte sich heraus, daß mit dem bei Nutzung eines turn-taking Mechanismus, mit einem Whiteboard (Tool 2 in Tabelle 2.4.5.1 und Abbildung 2.4.5.1) die Anzahl der niedergeschriebenen Ideen etwas niedriger lag, als bei dem Mindmappingtool (Tool 1). Mit dem parallel nutzbaren Whiteboard wurden etwa ein Viertel mehr Lösungen gefunden, als bei den Alternativen. Hieraus lässt sich schließen, daß eine parallele Arbeitsweise von Vorteil ist und bei der Anwendungsentwicklung für das System ermöglicht werden sollte. Dies sollte von den im System eingesetzten Anwendungen unterstützt werden.

Es kann zu synchronen oder asynchronen Abgleichungen mit dem System kommen. Abhängig davon, ob eine direkte Verbindung zum System besteht.

Die asynchrone Nutzung entspricht hierbei der Verwendung eines klassischen Versionierungstools. Benutzer checken ein Dokument aus, bearbeiten es und checken es wieder ein. Hierbei werden Veränderungen, die in keinem Konflikt mit konkurrierenden Änderungen stehen, direkt übernommen. Hat sich das Objekt in der Zwischenzeit verändert, werden die Stellen angezeigt, an denen Konflikte auftreten und gelöst werden müssen. Dies entspricht

einem Breakdown nach (Winograd und Flores 1987), da die Arbeit unterbrochen werden muss, um ein Problem zu beheben. Eine solche Vorgehensweise wirkt sich negativ aus, da der Benutzer seine Aufmerksamkeit auf ein neues Aufgabengebiet richten muss.

Die gleichzeitige Nutzung entspricht dem Normalfall während eines Meetings. Objekte werden auf Displays im Raum angezeigt und können von den Benutzern manipuliert werden. Hierbei werden Änderungen auf alle Displays verteilt. Dies sollte idealerweise so geschehen, daß nur wenige Kollisionen auftreten.

Durch die Anforderung des ortstransparenten Zugriffs können beide Arten der Nutzung vorkommen und müssen beim Design entsprechend berücksichtigt werden.

2.4.5.2. Operationen

Ein Objekt stellt anwendungsunabhängige und an den Anwendungsbereich angepasste Funktionen zur Verfügung. Die Anforderungen an spezielle Anwendungen können hier nicht untersucht werden, da sie abhängig vom speziellen Szenario sind und eigens betrachtet werden müssten.

Die anwendungsunabhängigen Operationen beinhalten die Manipulation von Metadaten und einen Satz allgemeiner Steuerbefehle.

Die Manipulation der Metadaten umfasst einerseits die Attributfelder wie Autor, Ort oder Stichwörter, aber auch das Verlinken mit anderen Dokumenten, die z. B. ein verwandtes Thema haben oder eine Anmerkung zur aktuellen Dokumentenstelle beinhalten.

Um solche Anmerkungen zu erstellen, muss es einfache Anwendungen geben, durch die der Inhalt der Anmerkung editiert werden kann. Was für Anmerkungstypen gewünscht sind, kann zwischen den Szenarios variieren. Zwei grundlegende Typen dürften jedoch einfache Texte und Bitmaps sein.

2.4.6. Verwaltung

Um das System auch für Dokumente mit höheren Sicherheitsanforderungen nutzbar zu machen, müssen die Objekte Informationen über die erlaubten Nutzungsmöglichkeiten enthalten. Das System muss ihre Einhaltung garantieren. Damit entsteht Bedarf nach Änderungsmöglichkeiten dieser Informationen und einer Benutzerverwaltung. Das Konzept und die Umsetzung dieser Rechte- und Benutzerverwaltung werden in (Mund 2006) beschrieben. Es muss gewährleistet werden, daß den Benutzern verschiedene Zugriffsrechte gewährt und entzogen werden können. Die Zugriffsrechte beziehen sich auf die erzeugten Inhalte. Durch

die Versionsverwaltung erweitert sich dies noch auf frühere und zukünftige Versionen. Auch der Zugriff auf bestimmte Geräte im Raum kann eingeschränkt sein.

2.5. Anwendungsfälle

Ausgehend von den im vorherigen Kapitel formulierten Anforderungen und den in (Neumann 2006) und (Burfeindt 2006) beschriebenen Szenarios, werden einige Anwendungsfälle behandelt. Einige ergeben sich aus der Natur des Systems als Mehrbenutzerumgebung und der Notwendigkeit einer Informationsverwaltung. Sie beschreiben die allgemeine Administration und die Recherche in Datenbeständen. Weitere Anwendungsfälle werden durch die gewünschten Eigenschaften des Systems erst möglich. Gewünscht wird die parallele Arbeit an Dokumenten sowie die Nutzung der verteilten Ausgabemöglichkeiten und der Anwendungskoordination. In den folgenden Abschnitten werden diese Anwendungsfälle beschrieben. Eine detaillierte Aufzählung findet sich in Anhang A.

2.5.1. Systemadministration

Das System ist für den Mehrbenutzerbetrieb ausgelegt und häufigen Konfigurationsänderungen unterworfen. Hieraus ergeben sich einige Aufgaben, die zur Verwaltung nötig sind. Diese unterscheiden sich größtenteils nicht von anderen Mehrbenutzertätigkeiten. Die Anwendungsfälle sind in Tabelle A.1 auf Seite 95 aufgeführt.

2.5.1.1. Benutzerverwaltung

Wie in jedem Mehrbenutzersystem muss es auch hier die Möglichkeit geben, Informationen über die Benutzer zu pflegen. Es müssen Benutzer angelegt und gelöscht werden können. Ihre Rechte und ihre Gruppenzugehörigkeit müssen an wechselnde Rollen angepasst werden. Manche Benutzer müssen vorübergehend gesperrt werden. Die verschiedenen Bereiche der Benutzerverwaltung sind delegierbar, so daß Gruppenleiter eigenständig agieren können. In Tabelle A.2 auf Seite 96 und Tabelle A.3 auf Seite 96 sind die Fälle des Anlegens und Löschens von Benutzern näher ausgeführt. Dies unterscheidet sich nicht vom Problem der Benutzerverwaltung in anderen Systemen und hängt noch vom verwendeten Rechtemodell und seiner Granularität ab.

2.5.1.2. Objektverwaltung

Objekte, die von einem Benutzer erstellt wurden, ebenso wie Systemobjekte, haben einen eingeschränkten Benutzerkreis mit unterschiedlichen Rechten. Wie bei der Benutzerverwaltung ist auch die Verwaltung der Objektrechte ein Thema, welches auch in anderen Systemen auftaucht. Welche Rechte und Beschränkungen dies sind, wird in der Diplomarbeit von Horst Mund (Mund 2006) behandelt. Es werden unter Umständen zusätzliche Rechte gebraucht, die in anderen Kontexten nicht nötig sind.

In diesem System umfasst die Objektverwaltung allerdings mehr als nur die Verwaltung der Zugriffsrechte. Hinzu kommt die semantische Anreicherung der Dokumente, d. h. das Speichern von Metadaten. Diese beschreiben den Inhalt und die Funktion der Dokumente. Eine solche Anreicherung wurde bereits in der Arbeit (Knop 2004) beschrieben, die ein persönliches Informationsverwaltungssystem zum Thema hatte. So kann die Effektivität einer Suche erhöht werden. Eine Zusammenstellung findet sich in Tabelle A.9 auf Seite 101.

2.5.1.3. Geräteverwaltung

Um Zugriffsbeschränkungen auf Raumebene einzuhalten, müssen die Geräte zu logischen Räumen zusammengefasst werden. Dort haben die Benutzer physischen Zugang zu jedem Gerät. Diese Gerät–Raum–Zuordnung sowie für einige Dienste der topologische Zusammenhang der Geräte müssen angepasst werden können. Dies kann auf mehrere Arten geschehen, je nach der erwarteten Häufigkeit von Konfigurationsänderungen. So kann diese Information in Konfigurationsdateien ausgelagert werden, die bei Änderungen editiert und neu geladen werden. Später kann eine Anwendung geschrieben werden, die es erlaubt, diese Informationen zu editieren. Wenn dem System genügend Informationen über die physikalische Anordnung der Geräte zur Verfügung stehen, kann das System solche Dateien selbstständig pflegen. Hierfür ist ein Weltmodell nötig. Beispiele für solche Weltmodelle sind das, in (Grossmann u. a. 2001) beschriebene, Nexus und das geometrische Weltmodell von EasyLiving (Brumitt u. a. 2000). Bei einer Erweiterung des Weltmodells ist die Umsetzung von locationbased Services. möglich.

2.5.2. Dienste nutzen

Die hier angesprochenen Anwendungsfälle finden sich in den Tabellen A.4, A.5, A.6 und A.7. Die Dokumente werden zusammen mit ihren Metadaten in einem logischen Zusammenhang gespeichert. Benutzer müssen Dokumente von Interesse finden können, da sie sonst nicht nutzbar sind. Zu diesem Zweck müssen mehrere Abfragemöglichkeiten vorhanden sein. Hier muss die Existenz von Metadaten genutzt werden. Neue Dokumente müssen beim Einfügen

in das System implizit mit möglichst vielen Metadaten versehen werden. Im System vorhandene Dokumente müssen nach dem Abrufen auch zu bearbeiten sein. Dabei können sich die Arten der Bearbeitung unterscheiden. Die einzelnen Diensten lassen sich benennen als:

- Dokumente finden
- Dokumente abrufen
- Dokumente bearbeiten
- Dokumente einfügen

2.5.3. Anzeige von Dokumenten

Das Anzeigen von Dokumenten in einem System mit mehreren Bildschirmen ist ein komplexes Thema, da auch die Übergänge möglichst einfach sein müssen. Die Anzeige sollte auch nach dem Verschieben auf ein entferntes Display beendet werden können. Hierzu sind mehrere Anwendungsfälle nötig. So muss die Anzeige eines Dokumentes auf einem beliebigen Display möglich sein und wieder geschlossen werden können. Eine Verschmelzung dieser beiden Fälle ist das Übertragen der Anzeige von einem Display auf ein anderes was auf verschiedene Arten geschehen kann. Die Tabelle A.8 auf Seite 100 enthält eine Aufstellung dieses Anwendungsfalls.

2.5.4. Geschichten

2.5.4.1. Präsentieren

Dieser Abschnitt beschreibt die Vorbereitung und den Ablauf einer Präsentation in den Begriffen der vorher definierten Anwendungsfälle. Zu Anfang muss das Dokument, das in der Präsentation verwendet werden soll, vom Vortragenden selbst oder einem Kollegen in das System eingefügt werden. Wenn noch Änderungen vorzunehmen sind, muss das Dokument aus der Datenhaltung abgerufen werden und die Änderungen eingepflegt werden. Kurz vor der eigentlichen Präsentation muss das Dokument abgerufen werden. Erst dann legt der Vortragende fest, auf welchem Gerät das Dokument angezeigt wird. Während des Vortrages muss die Anzeige den jeweils aktuellen Abschnitt fokussieren. Nach Abschluss des Vortrages wird die Anzeige beendet.

2.5.4.2. Teilnehmen an einer Sitzung

Im vorhergehenden Abschnitt wurde der Ablauf einer Präsentation aus der Sicht des Vortragenden geschildert. Im Gegensatz zu bisherigen Ansätzen sind die Zuhörer jedoch nicht auf Beteiligung durch Zurufe beschränkt, sondern können ihre Anmerkungen direkt an das gezeigte Dokument anhängen.

Der Benutzer wird registriert, wenn er den Raum betritt. Danach sieht er die Präsentationen anderer Benutzer. Auf seinem privaten Display kann er andere Ansichten der vorgestellten Objekte ansehen. Hat ein Objekt sein Interesse geweckt, kann er es sich vormerken, um später einfacheren Zugriff darauf zu haben. Kommen ihm während des Vortrages Ideen, kann er lokal neue Dokumente erstellen oder Anmerkungen zum besprochenen Dokument verfassen. Diese Bemerkungen unterliegen eigenen Zugriffsbeschränkungen, so daß sowohl private als auch öffentliche Anmerkungen möglich sind. Wenn das Treffen vorbei ist, meldet er sich am Raum ab, kann jedoch auch von außerhalb auf die Objekte zugreifen. Nur die Dienste, der im Raum installierten Geräte stehen ihm nicht mehr zur Verfügung. So kann er die markierten Objekte unterwegs bearbeiten. Solange keine Konflikte auftreten, werden seine Änderungen automatisch mit der zentralen Fassung abgeglichen. Andernfalls ist ein manueller Eingriff nötig. Bis der Konflikt manuell beseitigt wird, markiert ihn das System und pflegt eine parallele Version.

2.5.4.3. Aktives Teilnehmen

Falls ein Dokument nicht nur präsentiert werden soll, können die Teilnehmer den Inhalt auch aktiv mitgestalten. Die Anmeldung am Gerät und im Raum erfolgt analog zum letzten Abschnitt. Danach können sie das auf einem öffentlichen Display dargestellte Objekt bearbeiten. Die Teilnehmer verarbeiten ihre Änderungen parallel. Hierbei werden gleichzeitige Änderungen, die keinen Konflikt verursachen, zugelassen. Sollte ein Konflikt auftreten, wird eine Änderung zurückgezogen.

2.6. Zusammenfassung

Das System bietet mehrere Sichten auf ein Objekt. Diese Sichten können auch auf unterschiedlichen Geräten dargestellt werden. Der Umfang der von den Geräten zur Verfügung gestellten Funktionen kann variieren. Mehrere Benutzer sollen gleichzeitig auf den dargestellten Objekten arbeiten können. Auch sollen unterschiedliche Anwendungen koordiniert

arbeiten können, ohne von ihrer gegenseitigen Existenz zu wissen. (Anforderungen aus Abschnitt 2.3.2 und 2.3.3) Die vorhandenen Oberflächen der Clientgeräte sollen möglichst wenig verändert werden, aber mit den Funktionen, die im Bereich des Systems zur Verfügung stehen, erweitert werden.

3. Stand der Technik

Es gibt bereits einige Produkte, die versuchen den PC als Hilfsmittel für Gruppenarbeit zu etablieren. Häufig werden die individuellen Arbeitsplätze durch geteilte Datenhaltung, Grafikausgabe und verteilte Anwendungen erweitert. Diese Produkte konzentrieren sich jedoch in der Regel auf einen einzelnen Aspekt, so daß sie nicht umfassend genug für die Anforderungen sind. Daneben sind einige für die Verwendung in verteilten Umgebungen entworfen, so daß die Möglichkeiten räumlicher Nähe nicht genutzt werden. Zusammen mit Entwicklungen aus dem Ubiquitous Computing–Bereich bieten diese Produkte jedoch Bausteine, um die in den Anforderungen entworfene Umgebung zu realisieren.

In diesem Kapitel werden verschiedene Lösungen zur computergestützten Gruppenarbeit vorgestellt und ihre Merkmale mit den erarbeiteten Anforderungen verglichen. Es ist zu klären, ob ein eigenes System entwickelt werden soll und in welchem Ausmaß vorhandene Techniken übernommen werden.

Wenn räumliche Nähe der Teilnehmer vorausgesetzt wird, gibt es unterschiedliche Ansätze, Technik in Konferenzräumen gewinnbringend einzusetzen. Eine Möglichkeit ist das "Meeting Capture", bei der unauffällige Technologien eingesetzt werden, um die während eines Meetings erzeugten Artefakte und Ideen zu archivieren. Dies kann den Meetingablauf flüssiger gestalten, da mündlich vorgebrachte Ideen protokolliert werden. Es bringt jedoch keine neuen Möglichkeiten der Zusammenarbeit. So werden die Zeichnungen auf Whiteboards zwar gespeichert, sie sind jedoch nicht interaktiv. (Scholtz u. a. (2003), Richter u. a. (2001))

3.1. Verteilte Grafikausgabe

Hier gibt es verschiedene Lösungen, die sich in mehreren Merkmalen unterscheiden. Grob lassen sich drei Ansätze erkennen, wobei sich einzelne Lösungen teilweise mehreren Ansätzen zuzuordnen lassen. *Remote Desktop Software* kann die Arbeitsoberfläche des Benutzers auf einer entfernten Maschine darstellen und bietet die Möglichkeit, sich über das Netzwerk am Rechner zu identifizieren.

Supportsoftware kann ebenfalls Arbeitsflächen übertragen, beinhaltet aber keine Möglichkeit sich am entfernten Rechner anzumelden, sondern setzt eine vorhandene lokale Benutzersession voraus.

Neben Remote Desktop und Supportsoftware existieren noch Produkte für *Remote Collaboration*, die zur Unterstützung von Telekonferenzen entwickelt wurden. Diese Produkte bieten meistens noch zusätzliche Kommunikationskanäle, die die Verständigung bei Telekonferenzen erleichtern sollen. Dazu gehören z. B. eine Chatmöglichkeit oder Zeichenwerkzeuge. Allerdings muss der lokale Benutzer die Verbindung explizit veranlassen.

3.1.1. Remote Desktop

Remote Desktop Lösungen ermöglichen es, eine komplette Computerarbeitsoberfläche über ein Netzwerk darzustellen, nachdem man sich mit einem gültigen Benutzeraccount ausgewiesen hat. Hierbei sind auch mehrere unabhängige Benutzersessions möglich. Dieser Ansatz wird von den betriebssystemspezifischen Lösungen Apple Remote Desktop und Microsoft Remote Desktop sowie X-Windows implementiert. X-Windows bietet die Möglichkeit, die Oberfläche einzelner Anwendungen auf einem oder mehreren, entfernten Rechnern darzustellen. Wenn die Anwendung dies unterstützt, sind hierbei auch mehrere, parallele Eingabestreams möglich.

3.1.2. Supportsoftware

Der Hauptunterschied zu den Remote Desktop Lösungen ist, daß ein lokaler Desktop übertragen wird. Die entfernten Benutzer müssen eine gemeinsame Arbeitsfläche nutzen. Teilweise ist auch ein bereits angemeldeter lokaler Benutzer nötig. Vertreter dieser Gruppe sind verschiedene Umsetzungen des VNC-Protokolls[1] und PCAnywhere[2]. VNC ist ein relativ neues Protokoll, von dem in letzter Zeit viele Implementierungen aufgetaucht sind. Sie sind nicht nur für unterschiedliche Betriebssysteme erhältlich, sondern auch auf mobilen Clients, wie PocketPCs und Palm–PDAs. VNC ermöglicht auch parallele Verbindungen zur selben Benutzersitzung, wodurch man ein Display gemeinsam nutzen kann.

[1] VNC ist ein Protokoll zur Übertragung des Bildschirminhalts. Die Abkürzung steht für: **V**irtual **N**etwork **C**omputing

[2] `http://www.symantec.com/`

3.1.3. Remote Collaboration

Remote Collaboration Produkte existieren in peer-to-peer Versionen, in denen die Rechner der Benutzer die Kommunikation selbst verwalten. Hierbei ist die Größe der Konferenzen durch die Anbindung der Einzelplatzrechner begrenzt. Das Tool NetMeeting von Microsoft, das jedem Windowsbetriebsystem beiliegt, funktioniert auf diese Weise. NetMeeting bietet Zugriff auf Benutzerverzeichnisse sowie Sprachübertragung, textbasierten Chat, ein Whiteboard und das Übertragen von Anwendungsoberflächen. Diese können von jeweils einem Benutzer bearbeitet werden.

Die Beschränkung auf wenige Teilnehmer schuf die Voraussetzungen für Lösungen, die einen zentralen Server einsetzen. Dieser verwaltet und verteilt die Daten eines Meetings. Die Clients brauchen für eine Teilnahme nur noch geringe Ressourcen aufwenden. Beispiele für solche Lösungen sind Microsoft LiveMeeting (Microsoft 2006), Netviewer (Netviewer GmbH) und Bridgit (Smart Technologies). Alle Remote Collaboration Tools haben Funktionen die Übergabe der Kontrolle zu regeln, da die Kommunikation eingeschränkt ist. Es gibt keine parallele Bearbeitung von Dokumenten, mit Ausnahme eines Whiteboards.

3.1.4. Fazit

Bei allen Produkten wird die Darstellung abgebrochen, wenn der Rechner das Netz verlässt, der das Dokument enthält. Dies kann durch einen Absturz des Clientbetriebssystems oder Ausschalten passieren. Bei den serverbasierten Lösungen könnten solche Probleme vermieden werden. Außer der Darstellung werden keine Informationen geteilt. Nach dem Ende der Sitzung kann auf das Dokument nicht mehr zugegriffen werden. Wenn die Benutzer Daten teilen wollen, müssen sie dies explizit anstoßen. So kann ein Dokument, welches auf einem Teilnehmerrechner liegt, nur durch Blockieren des Teilnehmerrechners auf ein öffentliches Display gelegt werden. Ohne zusätzliche Anpassungen erkennen diese Programme die Anwesenheit neuer Partner oder öffentlicher Displays nicht. Diese Aufgabe und der Verbindungsaufbau müssen vom Benutzer übernommen werden.

3.2. Öffentliche Bildschirme

Öffentliche Bildschirme sind geteilte Computersysteme, die sich dadurch auszeichnen, daß ihre Bildschirme groß genug sind, um von mehreren Benutzern parallel genutzt zu werden. Sie ermöglichen es, Informationen in öffentlichen Räumen zu verbreiten. Wenn sie mit Eingabemöglichkeiten ausgestattet sind, kann eine kleine Gruppe auch gemeinsam an ihnen arbeiten.

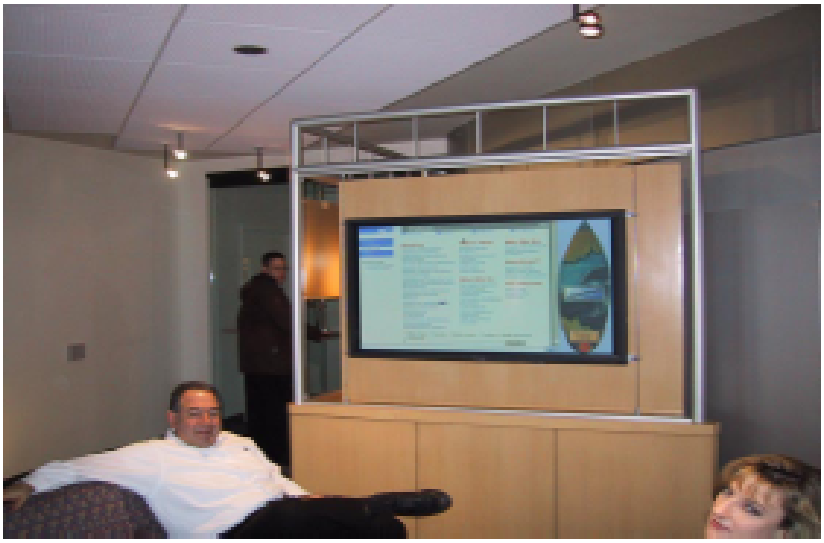

Abbildung 3.1.: Ein BlueBoard in die Einrichtung eines Pausenraumes integriert (Russell u. a. 2002a)

3.2.1. Smartboards

Smartboards sind berührungsempfindliche Whiteboards. Sie können an einen Computer angeschlossen werden. Auf diese Art können sie zum Archivieren des Inhalts genutzt werden. In Verbindung mit einem Videoprojektor können sie zur Bedienung des angeschlossenen Computers genutzt werden. Abgesehen von den Interaktionsmöglichkeiten der lokalen Benutzer, verändern sich die Eigenschaften des Rechners allerdings nicht. Eine Zusammenarbeit mit anderen Endgeräten ist nicht vorgesehen. Daher sind die Anforderungen des ortstransparenten Zugriffs auf Ressourcen und der Gemeinschaftsarbeit über das Netzwerk nicht erfüllt.

3.2.2. BlueBoard

BlueBoards sind große Touchscreens, die für einen schnellen Zugriff optimiert sind. Sie stehen an Orten, an denen sich Menschen spontan treffen. Sie sind mit Sensoren ausgestattet, die eine schnelle Identifikation erlauben. Hierdurch können Benutzer ohne Verzögerung die Funktionen nutzen, für die eine Identifikation notwendig ist, z. B. der Zugriff auf einen Homebereich und die Speicherung der behandelten Inhalte.

Die Identifikation erfolgt, indem eine Plakette mit einer eindeutigen Nummer vor ein Lesegerät gehalten wird. Mit dieser Plakettennummer wird in einer Benutzerdatenbank nach den

Abbildung 3.2.: Der persönliche Bereich eines BlueBoards. (Russell u. a. 2002a)

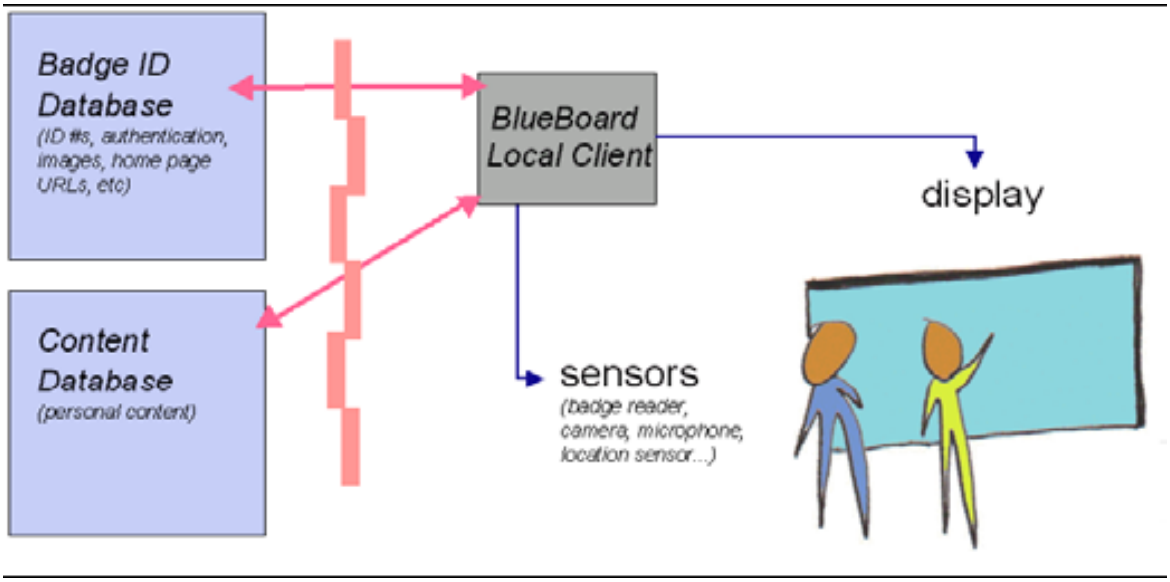

Abbildung 3.3.: BlueBoard verbindet die Inhalte eines Servers im Hintergrund mit einer Präsentation auf einem großen Plasmabildschirm. Dieser bietet die Möglichkeit, sich mit einer RFID–Marke auszuweisen. (Russell u. a. 2002b)

Benutzerinformationen gesucht. Sollten keine gültigen Benutzerinformationen gefunden werden, wird eine Fehlermeldung ausgegeben. Sobald ein Benutzer erkannt wurde, wird sein persönliches Icon (P-Con) dargestellt. Ein P-Con ist die Schnittstelle für persönliche Informationen. Will man Informationen dauerhaft teilen, zieht man sie von der öffentlichen Fläche auf das P-Con. Sollen eigene Inhalte gezeigt werden, werden sie vom P-Con auf die öffentliche Fläche gezogen. (Russell u. a. 2002b)

Neben der Möglichkeit Webseiten anzuzeigen, gibt es noch eine einfache Zeichenanwendung, mit der mehrere Personen synchron Inhalte erstellen können.

Das BlueBoard verbindet somit Inhalte von Servern im Hintergrund mit einer einfachen Präsentationsmöglichkeit. Der Aufbau eines BlueBoards wird in Abbildung 3.3 gezeigt. Der schnelle Zugriff auf Informationen ist das vorrangige Ziel. Das Anzeigen des eigenen Terminkalenders soll, inklusive Anmeldung, innerhalb von fünf Sekunden möglich sein. Ebenso einfach ist den das Teilen von erstellten Inhalten. (Russell u. a. 2002a, 2)

3.2.3. Fazit

Smartboards und BlueBoard sind als eigenständige Systeme gedacht, die nicht in eine weitere Umgebung integriert werden. Ein Smartboard kann allerdings zur einfachen Erweiterung der Fähigkeiten der Raumumgebung genutzt werden, indem die Möglichkeiten der lokalen Interaktion erweitert werden.

3.3. Gemeinschaftliche Anwendungen

Gemeinschaftliche Anwendungen beschränken sich meist auf einen Aspekt der Gruppenarbeit, den sie umsetzen. So gibt es kollaborative Zeichenprogramme, Whiteboardanwendungen, Texteditoren und Weiteres.

Zwei Beispiele für kollaborative Texteditoren sind Emacs und SubEthaEdit, welche mehreren Benutzern erlauben, Textdateien gleichzeitig zu bearbeiten. Die bearbeiteten Objekte haben bei beiden Produkten einen eindeutigen Eigentümer, auf dessen Rechner sie gespeichert sind. Die Editoren unterscheiden sich allerdings in der Art der Benutzereinbindung. Emacs ist eine zentrale Anwendung, die die X–Windowstechnik nutzt. Dadurch können weitere Benutzerschnittstellen auf entfernten Displays dargestellt werden. SubEthaEdit muss auf jedem Teilnehmerrechner gestartet werden. Der Editor, der die Datei lokal vorhält, synchronisiert nun die Anfragen der verbundenen Clients, um das Editieren zu ermöglichen. Dies hat den Vorteil, daß mehrere Dateien von mehreren Benutzern bearbeitet werden können, ohne daß es für die gesamte Arbeit einen Ausfallpunkt gibt. Ein fehlerhafter Client verhindert nur das

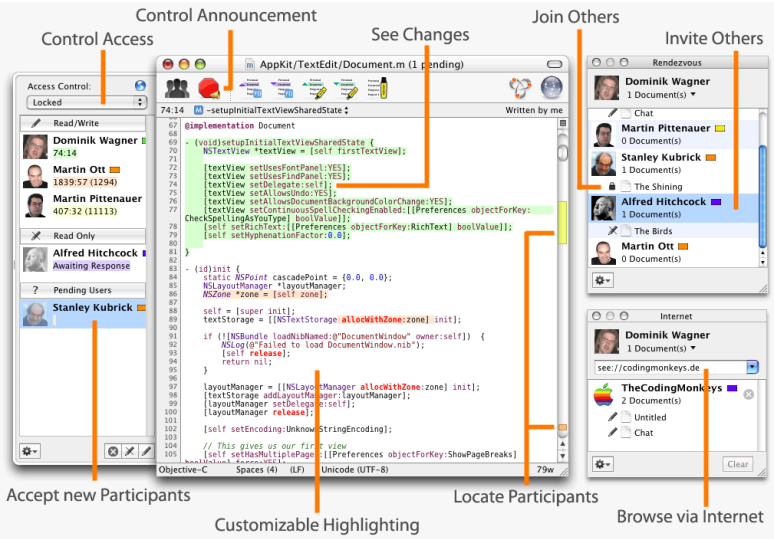

Abbildung 3.4.: Dieser Screenshot von (Wagner u. a. 2006) zeigt SubEthaEdit, einen Editor, der synchrones, kollaboratives Arbeiten erlaubt

weitere Bearbeiten von Dateien, die bei ihm gespeichert sind, allerdings auch dann, wenn die Datei für die anderen Clients im Netz erreichbar ist.

Bei SubEthaEdit kann der Besitzer eines Dokumentes andere Autoren einladen, die danach gleichwertigen Zugang zum bearbeiteten Dokument haben. Die aktuellen Position der anderen Benutzer und ihre Änderungen werden allen beteiligten Benutzern gezeigt.

Im lokalen Netzwerk werden offene Bearbeitungssitzungen automatisch angezeigt. Der Besitzer eines Dokuments sieht potenzielle Mitarbeiter. Außerhalb des LANs müssen mögliche Koautoren explizit nachfragen oder eingeladen werden. Es gibt allerdings zentrale Server, die die Koordinationsaufgabe übernehmen.

Beide Anwendungen unterscheiden sich in der Art der Benutzerkoordination. Unter Emacs öffnet sich, ausreichende Berechtigungen vorausgesetzt, direkt ein Fenster des Editors auf den Desktops der anderen Benutzer. SubEthaEdit nutzt die Technik Bonjour[3], um im lokalen Netz angebotene Files oder Koautoren darzustellen. Diese Statusanzeige erlaubt auch das initiieren von Kollaborationen. Wenn die Benutzer sich nicht im selben LAN befinden, gibt es die Möglichkeit zentrale Server zu nutzen, die diese Koordination anbieten. Alternativ kann die Verbindung durch Angabe einer IP–Adresse explizit aufgebaut werden.

Beide Beispiele sind für eine spezielle Dokumentenart konzipiert und können nicht erweitert werden. Zusätzlich hat Emacs keine Informationen über verfügbare Ressourcen. SubEthaEdit hat diese Möglichkeit und nutzt sie. Allerdings haben sich die Entwickler hierbei für eine

[3]Eine Technik zum spontanen Finden und Nutzen von Ressourcen. Von Apple entwickelt und Os X–spezifisch.

betriebssystemspezifische Lösung entschieden. Beiden Programmen fehlt eine Infrastruktur, um Files verteilt zu speichern und die Abhängigkeit von einzelnen Geräten zu reduzieren. Dies behindert die langfristige Zusammenarbeit im Team, da Dokumente ohne zentrales Repository nicht verfügbar sein können. Bei Verwendung eines zentralen Repositories können auch parallele Versionen der Dokumente entstehen.

3.4. Gemeinsam genutzte Datenspeicher

Repositories sind zentrale Datenspeicher, die neben den aktuellen Daten auch die vorhergehenden Versionen und eine Reihe von Metadaten speichern (Autor, Begründung, Datum und Zeit der Änderung, u.a.). Dies unterstützt die Benutzer beim Arbeiten mit einer gemeinsamen Datenbasis. Viele Entwicklungsumgebungen haben bereits eine Schnittstelle zu verbreiteten Versionsverwaltungen wie dem Concurrent Versions Systems (CVS) oder Subversion (SVN). Der Arbeitsablauf unterteilt sich hierbei in Phasen der Datensynchronisierung und des unabhängigen Bearbeitens. Nur in der Phase der Synchronisierung erfahren Benutzer Änderungen anderer Benutzer, wodurch nach jedem Synchronisieren Tests und Integrationsarbeiten notwendig werden. Um das Ausmaß der Konflikte überschaubar zu halten, müssen Synchronisierungen häufig stattfinden. Repositories sind größtenteils unabhängig von der Art der bearbeiteten Daten. Sie ermöglichen allerdings keine synchrone Gruppenarbeit. In ihrer gegenwärtigen Form sind sie nicht transparent, da Benutzer eine Synchronisierung explizit anfordern müssen. Wenn dies automatisiert wird, muss der Benutzer Konflikte, die durch Veränderungen seit der letzten Synchronisierung entstanden sind, eigenhändig auflösen.

Wenn eine transparente Schnittstelle zu Anwendungen möglich ist, deckt sich die Funktion der Repositories teilweise mit den Anforderungen an die Persistenzschicht.

3.5. Ubiquitous Computing Umgebungen

Andere Ansätze ersetzen die gewohnten Präsentationstechniken durch große, interaktive Displays, die einerseits weiterhin als Whiteboard genutzt werden können andererseits aber auch erweiterte Möglichkeiten der Zusammenarbeit bieten. Elektronische Dokumente können in unterschiedlichen Blickwinkeln und auf verschiedenen Displays gleichzeitig gezeigt werden. Die angezeigten Dokumente lassen sich direkt bearbeiten, wodurch eine Nähe zum Whiteboard entsteht. Es können weitere Aus-/Eingabegeräten im Konferenzraum integriert werden, die neue Interaktionsmöglichkeiten erschließen.

Diesen Ansatz verfolgen der iRoom ("Interactive room") der HCI Gruppe aus Stanford, die in (Johanson und Fox 2002) und (Russell u. a. 2005) beschrieben wird und die Roomware–Lösung von Ipsi Fraunhofer (Russell u. a. 2005), die im Rahmen des "I-Land"-Projekts entstand.

Die Installation von Roomware und dem iRoom ist teuer, da sie speziell entwickelte Hardware und Möbel nutzen, die teilweise noch nicht allgemein erhältlich sind. Die vorliegende Arbeit beschreibt die Nutzung handelsüblicher Hardware. So können einige Vorteile der interaktiven Konferenzraumumgebungen genutzt werden, ohne Investitionen in vergleichbarer Höhe zu erfordern. Die entwickelte Lösung soll allerdings auch die Geräte der beiden beschriebenen Lösungen integrieren können. Damit geht die erarbeitete in eine ähnliche Richtung wie Teamspace (Shih u. a. 2004), das auf der EventHeap–Technologie iROS aufbaut.

Beide Ubiquitous Computing Umgebungen verbinden Elemente aus den vorgestellten Lösungen. Meistens bieten sie noch Werkzeuge für Entwickler, die die Integration vorhandener Anwendungen erleichtern. Sie haben sich allerdings auf die Realisierung der Werkzeuge konzentriert, so daß der geforderte Sicherheitsaspekt nicht abgedeckt ist.

3.5.1. Roomware

Roomware ist ein Aspekt des I-Land Projektes der IPSI Gruppe der Fraunhofer Gesellschaft Darmstadt. Das Roomwarekonzept umfasst die Vernetzung aller, in einem Konferenzraum vorkommender Gegenstände. Außerdem ihre Erweiterung um Funktionen, die diese Vernetzung nutzen. Hierdurch soll die Inhaltsarbeit der Teilnehmer verbessert werden, da sie alle auf den gleichen Datenbereich zugreifen können.

3.5.1.1. Ausstattung

Das Fraunhoferteam hat für viele vorkommende Gegenstände eine Alternative entwickelt, die durch Integration eines Computers erweiterte Möglichkeiten zur Inhaltserstellung hat. Allen Alternativen gemeinsam ist die Bedienung durch berührungsempfindliche Displays und ein Netzwerkzugang. Durch die Stiftbedienung der Displays kann eine Nähe zum Arbeiten mit Papier erhalten werden. Die Liste der entwickelten Gegenstände umfasst:

ViewPort ein PDA-basiertes System

CommChair Ein Stuhl, in dessen Basis ein Computersystem eingebaut ist, das über ein schwenkbares Display steuerbar ist.

ConnecTable Ein Stehpult mit eingelassenem Display, das sich mit anderen ConnecTables koppeln lässt.

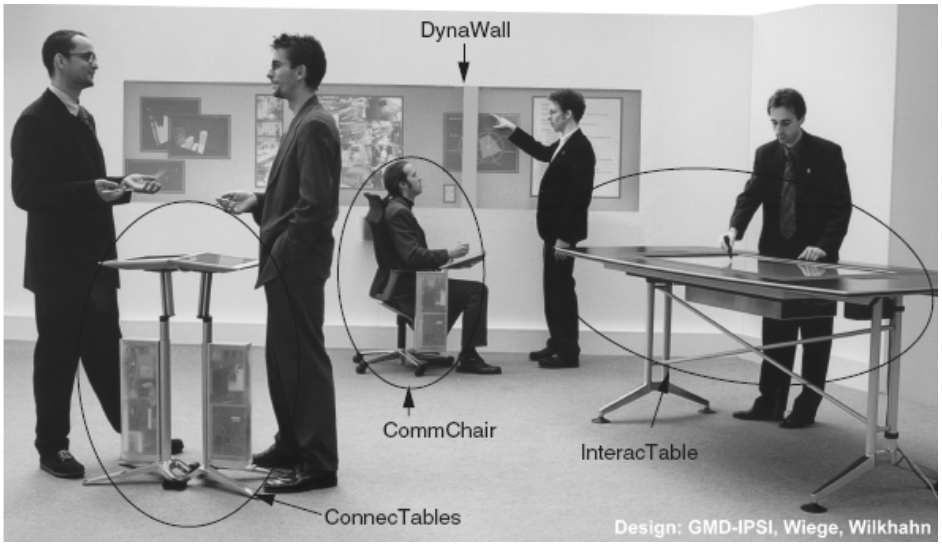

Abbildung 3.5.: Roomware – Second Generation((Tandler u. a. 2002))

DynaWall Ein großes Wanddisplay ($4 * 1$ Meter).

InteracTable Ein Tisch mit Platz für bis zu 6 Personen, welcher ein Touchscreen enthält.

Diese Komponenten wurden entworfen, um sich gegenseitig zu ergänzen und flexibel kombinierbar zu sein. Hierdurch können Teilnehmer miteinander arbeiten, die an verschiedenen Geräten stehen.

3.5.1.2. Technik

Unterschiedliche Geräte und Anwendungen müssen zusammenarbeiten, um effektiv zu sein. Das Roomwareframework versucht die Erstellung verteilter, kooperativer Software durch die Unterstützung hierfür nötiger Funktionen zu erleichtern. Es ist jedoch nicht auf die Integration vorhandener Anwendungen ausgelegt, sondern für die Entwicklung neuer Anwendungen.

Als Grundlage wurde das Framework COAST (Schümmer u. a. 2000) verwendet, welches Mechanismen zum synchronen Nutzen von Datenobjekten bereitstellt. Ausgehend von COAST wurde das Framework Beach (Tandler 2004) entwickelt. Hierbei ist COAST die Infrastruktur für die verteilte synchrone Bearbeitung von Objekten. Beach erweitert es um Funktionen für die Displaysteuerung und für Transformationen der Objektdarstellung. COASTs Möglichkeiten der verteilten Objekte wurden in eine API mit einer höheren Abstraktionsebene eingebettet. Beach ist mit dem Ziel entwickelt worden, als Grundlage für Anwendungen zur Ideenfindung zu dienen.

3.5.1.3. Anwendungen

Das Entwicklungsteam von Roomware hat einige Beispielanwendungen entwickelt, die zum Arbeiten in Gruppen genutzt werden können. Diese haben Ideenfindung und Organisation zum Thema. Normalerweise läuft eine Whiteboardanwendung, die dort ebenfalls nutzbar ist. Diese Anwendungen wurden in (Prante u. a. 2002) beschrieben und sind:

MagNet Eine Software die es erlaubt, virtuelle Karteikarten hierarchisch zu organisieren. Die Kanten der Karten ziehen sich magnetisch an.

PalmBeach Eine Anwendung für Roomware und PalmOs, die es erlaubt, unterwegs Karten für MagNet zu erstellen.

BeachMap Ein Mindmappingtool, das ebenfalls mehrere Benutzer und eine Zusammenarbeit mit PalmBeach unterstützt.

3.5.2. Interactive Workspaces

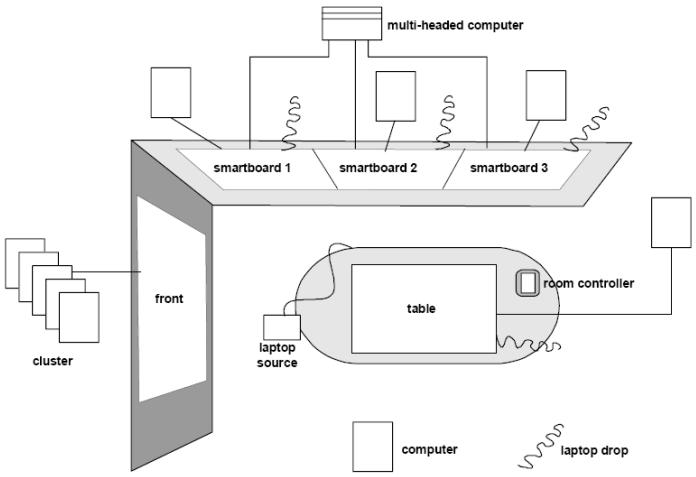

Abbildung 3.6.: Dieses Bild zeigt den Aufbau des iRooms. Dieser enthält ein hochauflösendes Display an der Stirnseite, drei Smartboards an einer Seitenwand, einem Touchscreen im Konferenztisch, sowie Netzanbindung für mobile Geräte. ((Johanson u. a. 2001, 2), Johanson und Fox (2002))

Die Forschungsgruppe HCI an der Universität Stanford (Stanford University, HCI Group 2005), unter der Leitung von Terry Winograd beschäftigt sich seit längerem mit neuen Formen der Computernutzung. Das Hauptziel der Gruppe ist es, Menschen bei der zwischen-

Abbildung 3.7.: Hier ist der iRoom in Benutzung zu sehen. Die in Abbildung 3.6 erwähnten Elemente sind auch hier zu sehen ((Johanson u. a. 2001, 2), (Johanson und Fox 2002))

menschlichen Interaktion zu unterstützen. Aus dieser Motivation heraus entstanden mehrere Projekte. Hierzu gehören der iRoom aus (Johanson und Fox 2002) und Teamspace (Shih u. a. 2004).

3.5.2.1. iRoom

Der iRoom ist ein Prototyp einer Umgebung mit ubiquitären Computern und enthält ähnliche Ausstattungsmerkmale wie die I-Land Umgebung des Fraunhoferinstituts. Ziel des Projektes war es, ein Framework zu schaffen, das die Entwicklung ubiquitärer Anwendungen erleichtert. Hierfür wird die Kooperation zwischen heterogenen Systemen auf eine höhere Abstraktionsebene verlagert.

Die Schnittstelle des iRooms besteht aus mehreren Geräten, die äußerlich eine große Ähnlichkeit mit den Komponenten von Roomware haben. Der iRoom enthält drei digitale Whiteboards, ein hochauflösendes Display an der Stirnseite und einen Tisch mit einem von unten projizierten Bild. Die Teilnehmer bringen auch persönliche Notebooks und mobile Geräte mit, die ebenfalls integriert werden.

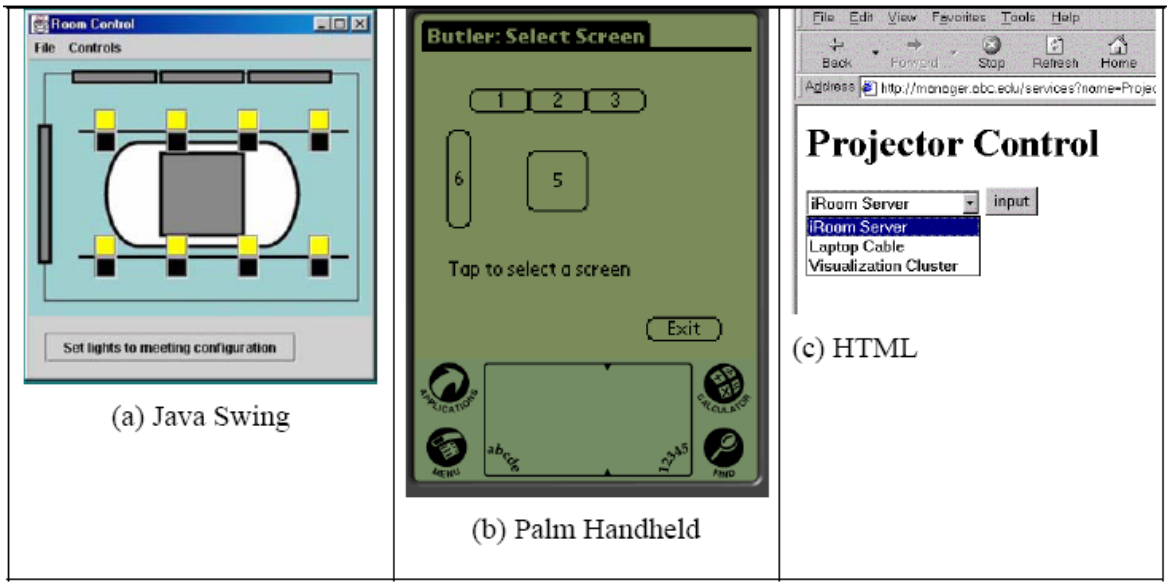

Abbildung 3.8.: Dieselbe Displaysteuerung in Swing, als Palmapplikation und als Webseite

3.5.2.2. iROS — Das Betriebssystem des iRoom

Um ihr System zu realisieren entwickelte das Team des iRoom das “MetaOs” iROS, daß die grundlegende Funktionalität zur Verfügung stellt.

Es wurden hauptsächlich drei Arbeitsweisen ausgemacht, die unterstützt werden sollten:

- Gemeinsames Benutzen von Displays
- Orts– und Typenunabhängige Datenspeicherung
- Koordination von Anwendungen

Um diese Funktionen mit möglichst geringem Aufwand, von alten Anwendungen aus, nutzbar zu machen, wurden drei Technologien entwickelt. Sie stellen keine Hilfsmittel für paralleles Arbeiten zur Verfügung, sondern konzentrieren sich auf die Integration von alten und neuen Anwendungen in eine "Ubiquitous Computing"-Umgebung.

EventHeap Eine Kommunikationsinfrastruktur, welche Anwendungen stark entkoppelt.

DataHeap Ein Datenspeicher, der Daten nur durch Indizes auf ihren Metadaten zugreifbar macht und durch ein Transformationsframework Typunabhängigkeit erreicht.

iCrafter Gibt Services bekannt und stellt im Bedarfsfall einfache Controller zur Verfügung.

3.5.2.3. Anwendungen im iRoom

Als erste Beispielanwendung, die diese Möglichkeiten nutzen sollte, wurde Smart Presenter entwickelt. Dieses Präsentationsprogramm koordiniert die Darstellung von Objekten auf mehreren Displays. Hiermit kann eine Präsentation erstellt werden, die die Vorteile von mehreren Darstellungsarten kombiniert.

3.5.2.4. Teamspace

Teamspace ist eine "Ubiquitous Computing"–Umgebung mit dem Ziel, die Zusammenarbeit von studentischen Arbeitsgruppen zu erleichtern. Es basiert auf dem "Meta OS" iROS und bietet die Möglichkeit, auf einem geteilten Display kooperativ zu arbeiten.

Teamspace–Installationen bestehen aus einem festinstallierten Rechner, der ein großes Display steuert und als kooperative Arbeitsfläche anbietet.

Haben sich die Benutzer durch Eingabe eines auf dem öffentlichen Display dargestellten Codes authentifiziert, können sie gemeinsam arbeiten. Die Möglichkeiten hierbei sind:

- Dateitransfer zwischen Teilnehmern
- Darstellen eines Dokuments auf verschiedenen Displays
- Steuerung der gemeinsamen Arbeitsfläche

3.5.3. Fazit

In den vorhergehenden Abschnitten wurden beispielhaft zwei Umgebungen für ubiquitäres Computing vorgestellt. Beide haben ähnliche Ziele, wie die in dieser Arbeit formulierte Vision. Die Unterstützung von Menschen bei Meetings und der Nutzung entsprechender Technik. Hierbei hat sich iROS darauf konzentriert, Anwendungen in eine "Ubiquitous Computing"–Umgebung zu integrieren. Roomware hat eine Infrastruktur für die Entwicklung verteilter Anwendungen mit parallelen Bearbeitungsmöglichkeiten zur Verfügung gestellt.

Beide Dienste haben keine Unterstützung für die dauerhafte Speicherung von Dokumenten, sondern bieten stattdessen einen Datenspeicher an, der während ihrer Laufzeit Dokumente verfügbar hält. Roomware enthält bereits die Möglichkeit, Benutzer zu identifizieren. Weitergehende Sicherheitsfeatures wurden jedoch in der Beschreibung der Infrastruktur ausgeklammert (Tandler 2004, 214). iROS ist grundsätzlich auf Offenheit und Integration ausgerichtet. Der Prototyp iRoom ist nur gegen Zugriff von außen gesichert. Teilnehmer haben alle die gleichen Rechte. Ein Grund, der dafür in (Johanson u. a. 2001, 12) angegeben wird, ist

das Fehlen eines angemessenen Sicherheitsmodells für kollaborative Umgebungen. Dazu kamen noch die unterschiedlichen Sicherheitsannahmen der benutzten Plattformen.

3.6. Zusammenfassung

Alle Techniken erfüllen nur einen Teil der Anforderungen. Die größte Abdeckung besteht naturgemäß bei den "Ubiquitous Computing"-Umgebungen, die ein ähnliches Szenario haben. Die verschiedenen Produkte können daher in Teilbereichen integriert werden, stellen jedoch keine fertigen Lösungen dar. Roomware ist bereits ein weiterentwickeltes Produkt, das jedoch nur in Teilen verfügbar ist und als geschlossenes System eine Neuentwicklung sämtlicher Produktivitätstools erfordert. iROS erfüllt ebenfalls nicht alle Forderungen, ist jedoch im Sourcecode verfügbar und sehr offen ausgelegt. Daher sollte man ältere Programme und neue Techniken relativ leicht integrieren können.

4. Design

In Kapitel 2 wurden die Anforderungen und das Ziel des Systems beschrieben. Danach ist in Kapitel 3 untersucht worden, ob vorhandene Produkte die Anforderungen bereits erfüllen. Da kein Produkt mit den formulierten Anforderungen übereinstimmte, wird nun eine eigene Lösung entworfen. Am Anfang stehen hierbei grundlegende Entscheidungen zur Aufteilung der gewünschten Funktionen. Daraufhin wird die Architektur der Kommmunikationskomponente beschrieben. Danach folgen die Beschreibungen weiterer Dienste und der generalisierten Schnittstellen, die von Anwendungsentwicklern genutzt werden können. Zum Abschluss folgt die Beschreibung des generellen Aufbaus von integrierten Anwendungen und zusätzlicher Komponenten, welche die Systemdienste übernehmen.

4.1. Annahmen und Eingrenzungen

Der nachfolgende Entwurf findet unter der Bedingung statt, daß mindestens die folgenden Geräteklassen zur Verfügung stehen:

Öffentliche Displays Sie bestehen aus einem Rechner mit einem ausreichend großem Display, so daß es mehrere Personen nutzen können. Ein Videoprojektor am Grafikausgang würde diese Bedingung erfüllen.

Endgeräte mit Tastatur, Maus und Bildschirm Dies wären z. B. Laptops oder Arbeitsplatzcomputer.

Server für Infrastruktur

An die verwendeten Betriebssysteme werden keine außergewöhnlichen Anforderungen gestellt. Sie müssen lediglich Multiuserfähig sein und Unterstützung für Netzwerke bieten.

Es wird vorausgesetzt, daß diese Geräte untereinander TCP/IP Verbindungen aufbauen können, um Daten auszutauschen. Dies geschieht je nach Gerät und Mobilität drahtgebunden oder drahtlos.

Zusätzlich sind die Benutzer langfristig beteiligt und nutzen das System beruflich. Dadurch ist es möglich, eine vorherige Registrierung und eine kurze Einführung der Benutzer vorauszusetzen. Die benötigte Zeit sollte sich in begrenztem Rahmen halten, da die fachlichen

Aufgaben im Vordergrund bleiben sollen. Eines der Ziele des Systems ist es, aus der Wahrnehmung des Benutzers zu verschwinden. Eventuell kann die Einführung durch selbsterklärende Elemente und soziales Lernen ersetzt werden.

4.2. Grundlegendes

In diesem Abschnitt soll die grundlegende Struktur des Systems und ihre Herleitung beschrieben werden. Die ersten Architekturentwürfe beschreiben das System und die daraus folgenden Entscheidungen als Ganzes. In späteren Abschnitten werden einzelne Aspekte tiefergehend behandelt.

4.2.1. Außensicht

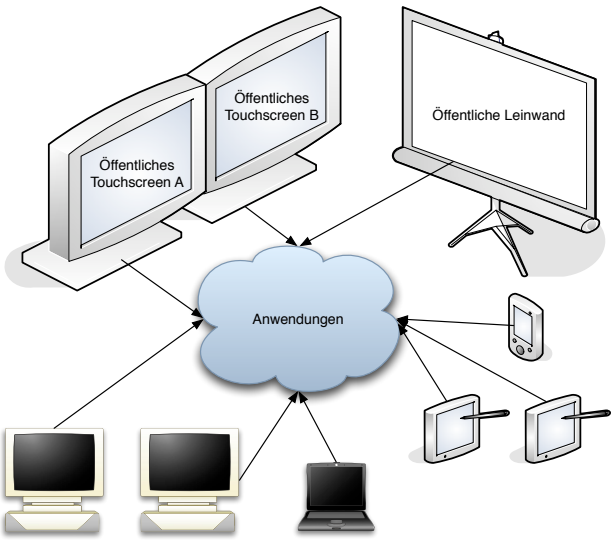

Abbildung 4.1.: Aus Benutzersicht verhält sich der ganze Raum wie ein System mit mehreren Ein– und Ausgabegeräten.

Als Ganzes betrachtet, besteht das System aus einer Sammlung von Ein– und Ausgabegeräten, die über eine Blackbox verbunden sind. Interaktion mit Dokumenten ist hierbei nicht von Gerätegrenzen abhängig, diese sind für den Benutzer transparent. Für diese Transparenz müssen die unterschiedlichen Geräte miteinander verbunden sein und das Wissen über ihre inneren Zustände verbreiten. Diese Situation ist in Abbildung 4.1 dargestellt, bei der die

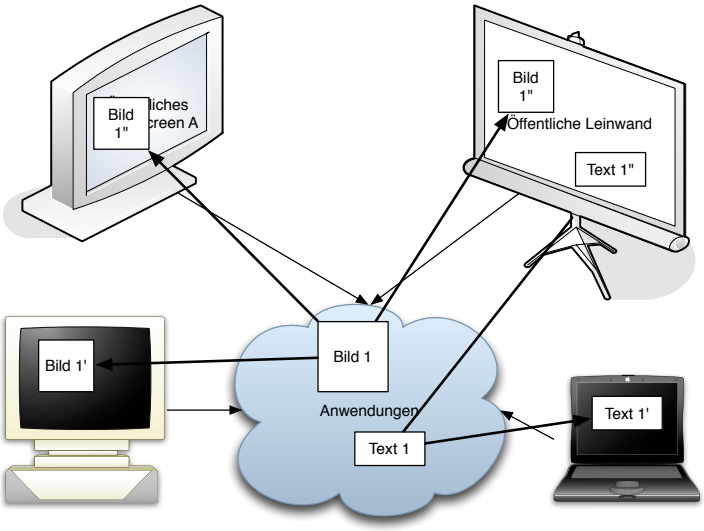

Abbildung 4.2.: Objekte im System werden in mehreren Sichten dargestellt, die koordiniert werden müssen.

Geräte mit Benutzerschnittstellen klar definiert sind, während die Art des Gesamtsystems unklar bleibt.

Einzelne Dokumente sind im System vorhanden und können mit unterschiedlichen Sichten dargestellt und bearbeitet werden. Als Beispiel zeigt Abbildung 4.2 ein Text– und ein Grafikdokument, die parallel auf mehreren Geräten bearbeitet werden. Eingabegeräte sind hierbei nicht fest mit einem Bildschirm verbunden, sondern können ihren Eingabefokus wechseln, so daß auch entfernte Displays mit Maus und Tastatur bedient werden können. Um dies zu erreichen, müssen die Benutzereingaben, ebenso wie hierdurch ausgelöste Aktionen bekanntgegeben werden. Dadurch stellt das System einen zusammenhängenden virtuellen Raum dar, der über die verschiedenen Geräte betrachtet und manipuliert werden kann.

4.2.2. Aufteilungsmöglichkeiten

Es gibt mehrere Arte das System zu strukturieren, um dieses Verhalten zu erreichen. Man benötigt hierfür eine Möglichkeit zur Interprozesskommunikation und Mehrbenutzerfähigkeit. Mehrbenutzerfähige Anwendungen werden heutzutage häufig als Client/Server–Architektur ausgeführt. Ein Extremfall des Client/Server–Modells sind Terminalsysteme. (Abbildung 4.4

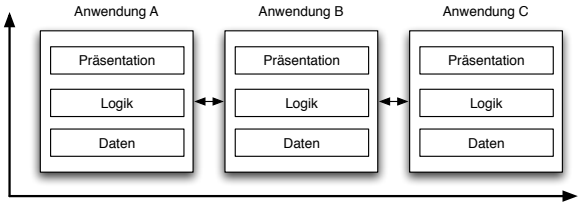

Abbildung 4.3.: Die Kommunikationsanforderungengliedern sich in zwei unterschiedliche Bereiche: Innerhalb einer Anwendung und zwischen unterschiedlichen Anwendungen

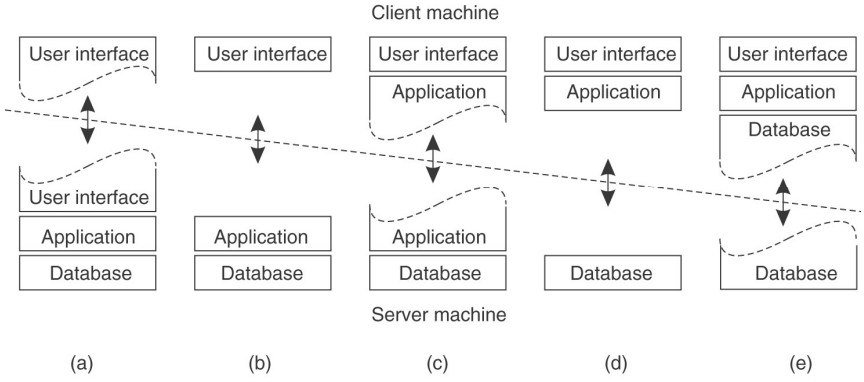

Abbildung 4.4.: Innerhalb einer Anwendung muss entschieden werden, wie die einzelnen Funktionen im Netz verteilt werden. (Tanenbaum und van Steen 2002)

a) Bei dieser Art Trennung reicht die Verteilung der Funktionalität von Terminals [1] bis zu Fat Clients [2].

Da für die Erfüllung der Anforderungen eine Kommunikation zwischen den Applikationen nötig ist, ergibt sich eine weitere, vertikale Trennlinie. Eine Aufteilung dieser Art ist in Abbildung 4.3 dargestellt. Es widerspricht dem Ziel eines wartbaren Systems, wenn getrennte Aufgabenbereiche von derselben Anwendung angeboten werden (Kahlbrandt (2001), Raasch (1992)). Aufgabenbereiche, die nicht eng miteinander verwandt sind, sollten in getrennten Applikationen umgesetzt werden. Sie kommunizieren dann über definierte Schnittstellen. Dies entspricht dem Prinzip der Modularisierung und Kapselung, durch Aufteilung in getrennte Einheiten, deren Innenleben nicht bekannt ist.

4.2.3. Zentraler Server oder verteiltes System

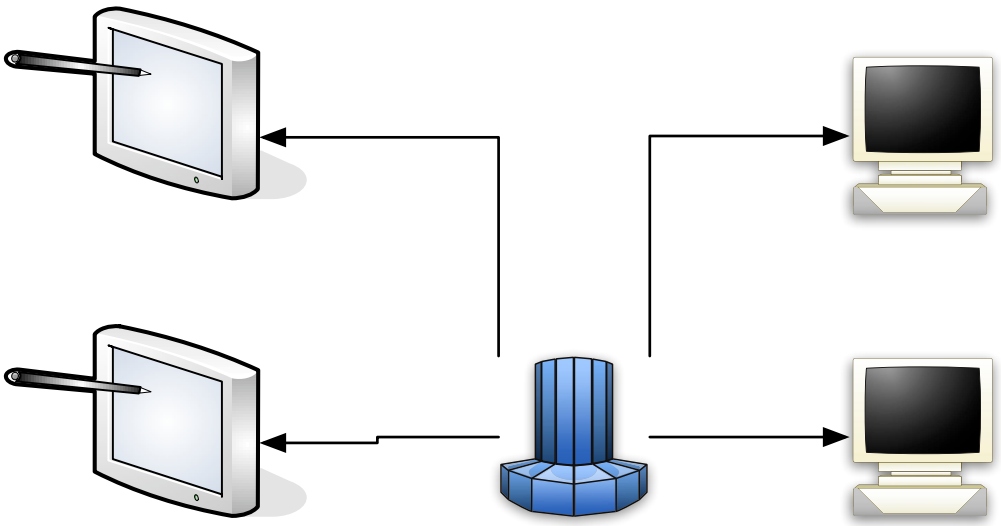

Abbildung 4.5.: Zentraler Server mit Terminals

Eine vorstellbare Architektur wäre nun ein zentraler Server, der sämtliche Applikationen ausführt und die Benutzerschnittstellen auf Terminals darstellt. Dies entspricht Punkt a oder b in Abbildung 4.4. Diese in Abbildung 4.5 dargestellte Architektur verlagert die Aufgabe der Benutzerkoordination in eine zentrale Anwendung. Eine solche Anwendung muss jedoch das Darstellen und Koordinieren mehrerer Sichten und Bearbeitungskontexte unterstützen. Anwendungen mit diesen Eigenschaften sind selten und müssten neu entwickelt werden.

[1] Clients stellen das User–Interface dar (a)
[2] Nur ein Teil der Datenhaltung liegt auf dem Server; Spalte d

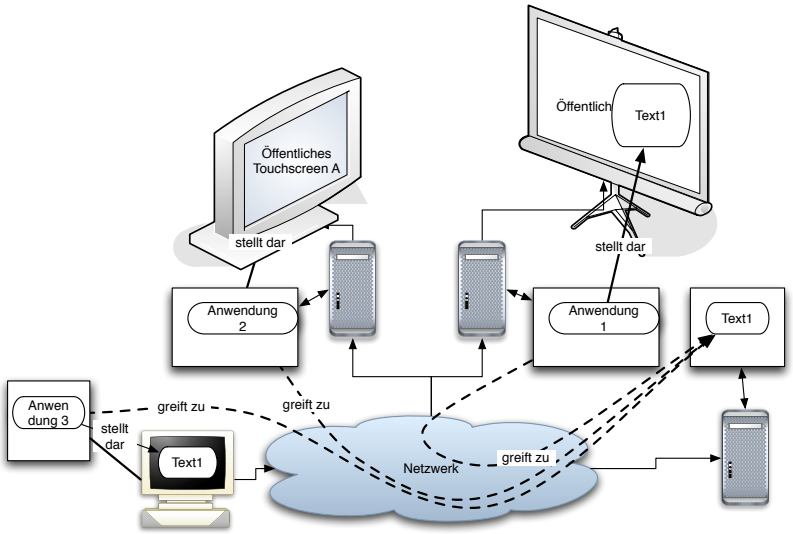

Abbildung 4.6.: Jedes Display wird von einem Rechner gesteuert, der auch Anwendungen ausführt.

Es müsste außerdem noch eine Komponente zum Nachrichtenaustausch zwischen den Anwendungen integriert werden wie in Abbildung 4.3 illustriert. Ein Vorteil wäre die geringe Nachrichtenlaufzeit bei der Kommunikation zwischen den einzelnen Komponenten.

Eine bekannte Klasse zentraler Programme mit parallelem Mehrbenutzerbetrieb sind Datenbanken, für die es bereits häufig untersuchte Techniken zur Konfliktvermeidung und Erkennung gibt. Diese Techniken müssten, bei Verwenden eines zentralen Servers, von jeder angebotenen Anwendung umgesetzt werden. Dies erschwert die Integration von vorhandenen Anwendungen, da sie selten mehrere Sichten bieten. Damit müssten sie je Display einmal ausgeführt werden.

Der Implementierungsaufwand einer Terminalserverlösung ist hoch, bedingt durch die Neuentwicklung der Anwendungen. Es kommen die Kosten für einen leistungsstarken, zentralen Server hinzu. Sollte der Rechenleistungsbedarf steigen, stellt sich die Frage der Skalierbarkeit, die durch den Typ des zentralen Servers begrenzt ist.

Im Folgenden wird eine Alternative betrachtet, die Applikationslogik und Benutzerschnittstelle auf den Endgeräten ausgeführt. Es gibt eine zentrale Datenhaltung, die die konkurrierenden Zugriffe verwaltet. Dieser Aufbau hat den Vorteil, daß Verzögerungen in der Darstellung geringer sind, da die Benutzereingaben und die Oberflächenbeschreibung nicht über das Netz transportiert werden müssen.

Hierbei sind die Applikationen eigenständige Einheiten auf jedem Endgerät, die über defi-

nierte Schnittstellen aus dem Netz angesprochen werden können. Um den Anwendungen, trotz Unterschieden in den zugrunde liegenden Plattformen, eine gemeinsame Möglichkeit zur Kooperation zu bieten, wird eine Middleware verwendet. Die einzelnen Komponenten des verteilten Systems werden durch diese Middleware betriebssystemunabhängig verbunden.

Sowohl bei einer zentralen, als auch bei einer verteilten Lösung, müssen die Applikationen kommunizieren, um ihr Verhalten zu synchronisieren. Die Problematik synchroner Zugriffe ist ebenfalls in beiden Ansätzen zu lösen. Man kann die Lösung dieser Probleme jedoch teilweise vor den Applikationen verbergen, wenn man ihnen eigene Schnittstellen zu wichtigen Funktionen bietet. Die Komponente, die diese Schnittstellen implementiert, kapselt nun die Details der Zugriffe. Dadurch kann man auch Altanwendungen mit geringem Aufwand integrieren. Da ein Verbund von unabhängigen, aber zusammenarbeitenden Rechnern leichter skalierbar ist als ein zentraler Server, wird die Architektur als verteiltes System vorgesehen.

4.3. Aufteilung des Systems

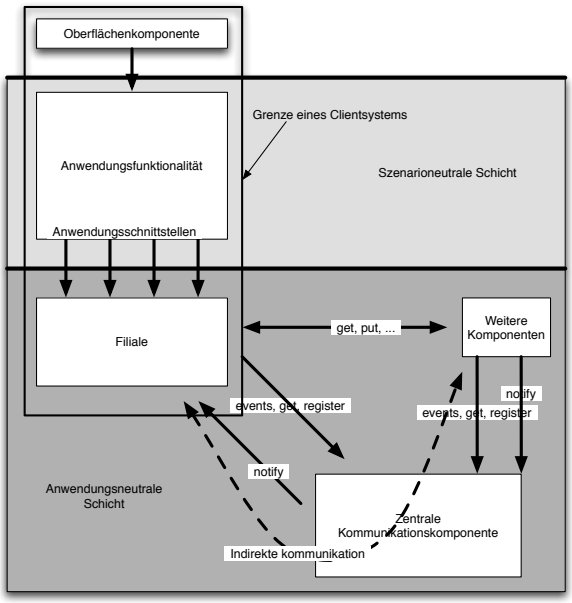

Abbildung 4.7.: Das System wird in drei Schichten geteilt: Anwendungsunabhängig, szenariounabhängig und szenariospezifisch

Abbildung 4.7 zeigt eine mögliche Aufteilung des Systems. Die anwendungsneutrale Filiale implementiert die Anwendungsschnittstellen zu den anderen Komponenten des verteilten Systems. Anwendungen nutzen Schnittstellen, um mit anderen Komponenten zu kommunizieren, soweit dies für ihre Steuerung und das Bereitstellen der Anwendungsfunktionalität nötig ist. Darüber liegt die an das gewünschte Einsatzszenario angepasste Oberflächenkomponente. Diese greift auf die vorhandenen Anwendungen zurück. Durch die Definition von Schnittstellen zwischen Oberflächenkomponente und Anwendungsfunktionalität können die beiden Komponenten durch verschiedene Implementationen ersetzt werden.

Es wird ein Satz von Schnittstellen definiert, der die Steuerung der Anwendung, soweit wie möglich, einheitlich festlegt. Diese Schnittstellen sollen die Koordination verschiedener Anwendungen vereinfachen. Auch zur Kommunikation zwischen Anwendung und Middleware existieren Schnittstellendefinitionen, die die Voraussetzungen beschreiben, um als Teil des Gesamtsystems zu fungieren.

Vorhandene Anwendungen werden diese Voraussetzungen nicht erfüllen, so daß Adapter notwendig sind, die das Interface der Anwendung kennen und in das Interface der Middleware übersetzen. Ein Entwurfsmuster für Adapter ist in (Gamma u. a. 1994, 139-150) beschrieben. Zwischen Benutzerschnittstelle und Anwendungsfunktionalität ist ebenfalls ein Adapter notwendig, z. B. um allgemeine Fenstermanagerfunktionalität zu übernehmen. In so einem Fall würde dieser Adapter auch direkt die Middleware ansprechen[3].

Das Gegenstück dieser Schnittstellen wird von einer lokalen Komponente der Middleware implementiert. Diese übernimmt generelle Infrastrukturaufgaben, die normalerweise vom Betriebssystem erfüllt werden oder nur in einem verteilten System nötig sind. Damit verbergen sie spezifische Details der verschiedenen Netzwerkprotokolle und den unterschiedlichen Umsetzungen der Systemaufrufe. So kann in diesem verteilten System auch die Betriebssystemebene heterogen sein.

Die Schnittstellen, die von Anwendung und Middleware implementiert werden müssen, werden in Abschnitt 4.12 definiert.

Die lokale Komponente kommuniziert mit fremden Systemen und nutzt einen zentralen Kommunikationsservice, um diese zu finden. Weitere Komponenten, die einen Sonderstatus innehaben, sind der Persistenzserver und eine Sicherheitskomponente. Hierbei verwaltet der Persistenzserver den Zugriff der Anwendungen auf Datenobjekte und die Sicherheitskomponente überwacht den Zustand des Systems. Sie erteilt oder verweigert Anwendungen die Berechtigung für bestimmte Aktionen.

[3] Bei Übertragen der Darstellung auf ein anderes Display

4.4. Struktur des Verteilten Systems

Für die Organisation und den Entwurf verteilter Systeme gibt es bereits mehrere bewährte Architekturen, die sich in ihren Zielsetzungen teilweise unterscheiden. Die Architekturen verwenden unterschiedliche Techniken, um die Kommunikation zwischen verteilten Objekten zu regeln. Dabei handelt es sich um entfernte Methodenaufrufe, Nachrichtenaustausch und Datengetriebene[4] Ereignissteuerung. Es werden kurz zwei der Techniken vorgestellt und ihre Eigenschaften sowie Vor- und Nachteile in Bezug auf die Anforderungen beschrieben. Für jede der hier vorgestellten Techniken existieren bereits fertige Architekturen, die auch für verschiedene Plattformen implementiert wurden. So ist Corba ein Beispiel für Remote Method Invocation, Blackboard hingegen eine frühe Umsetzung eines datengetriebenen Ansatzes.

4.4.1. Methoden und Funktionsaufrufe

In Abbildung 4.8 sind zwei kommunizierende Anwendungen skizziert. Es wird das Übergeben eines Filehandles und das Auslösen einer Funktion Dargestellt. Um das zu ermöglichen, muss die angesprochene Anwendung bereits im Hintergrund laufen. Es gibt mehrere Frameworks, die das Entwickeln von Client–Server–Anwendungen unterstützen, wie z. B. Corba oder RMI. Beide haben bereits Möglichkeiten zum Entdecken von laufenden Diensten.

Bei diesem aufrufbasierten Modell gehen Client- und Serveranwendung eine starke Bindung ein, da die Clientanwendung im Voraus wissen muss, wen sie beauftragt. Eine Anwendung muss, bis zur Abarbeitung der Methode, eine Verbindung aufrecht erhalten. Da es in dieser Umgebung zu spontanen Ausfällen von Dienstanbietern kommen kann, muss nach einer anderen Lösung gesucht werden.

4.4.2. Data Driven

Eine Alternative zum direkten Aufrufen ist ein ereignisgesteuertes Modell. In diesem erzeugen Programme Ereignisse, die Informationen über erwünschte Effekte oder eingetretene Statusänderungen enthalten. Andere Anwendungen laufen als Daemons und lesen das Ereignisprotokoll mit. Erkennen sie ein Ereignis, das ihre Dienste beschreibt, führen sie die gewünschte Aktion aus und erzeugen ein neues Ereignis, in dem die Ergebnisse stehen. Benutzer und Anbieter mussten zu keinem Zeitpunkt gegenseitig voneinander wissen, sie brauchten nur ein gemeinsames Medium um die Ereignisse zu veröffentlichen.

Eine Umsetzung ist ein Abonnement–Server, bei dem Anwendungen ihre Interessen anmelden und benachrichtigt werden, wenn ein Ereignis ihre Konditionen erfüllt. Dies entspricht

[4] data driven

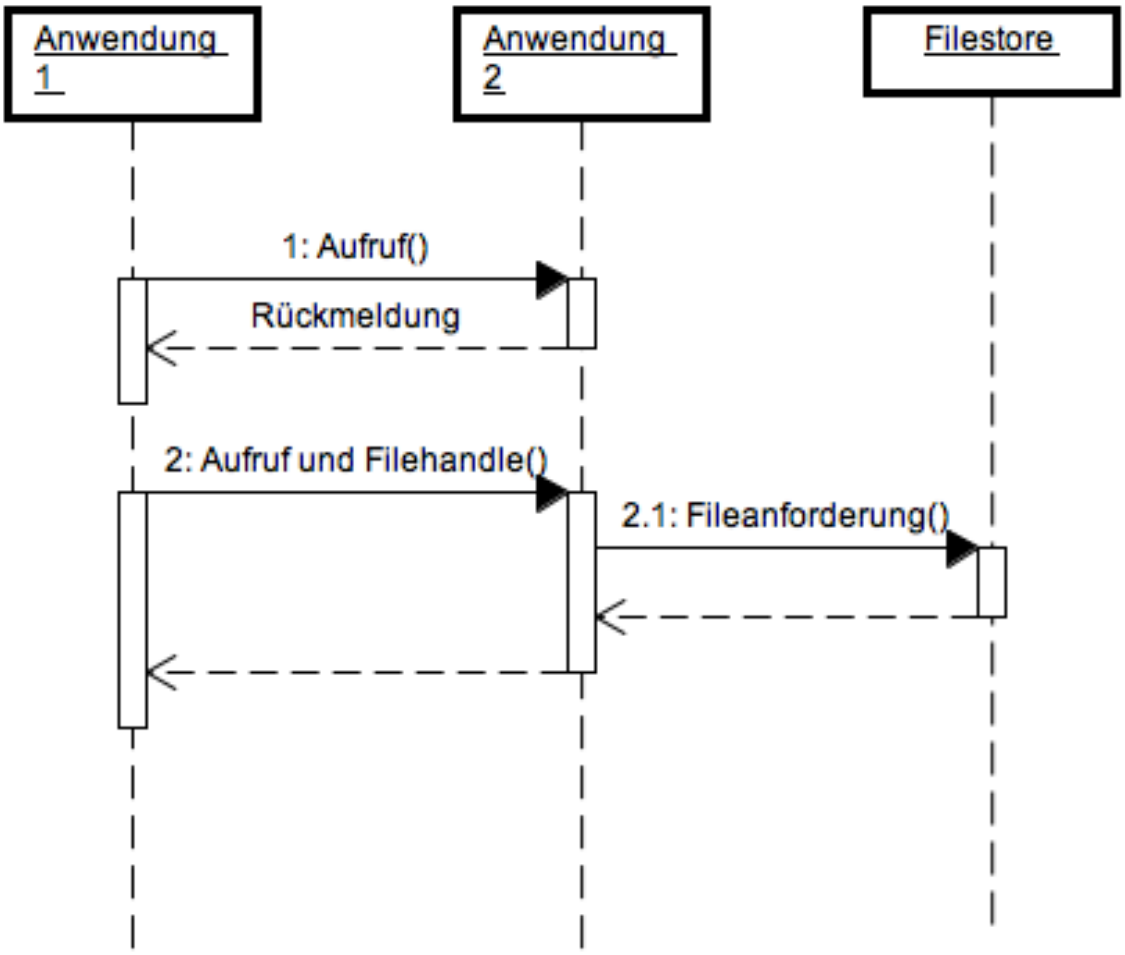

Abbildung 4.8.: Verschiedene Anwendungen und Kollaboration

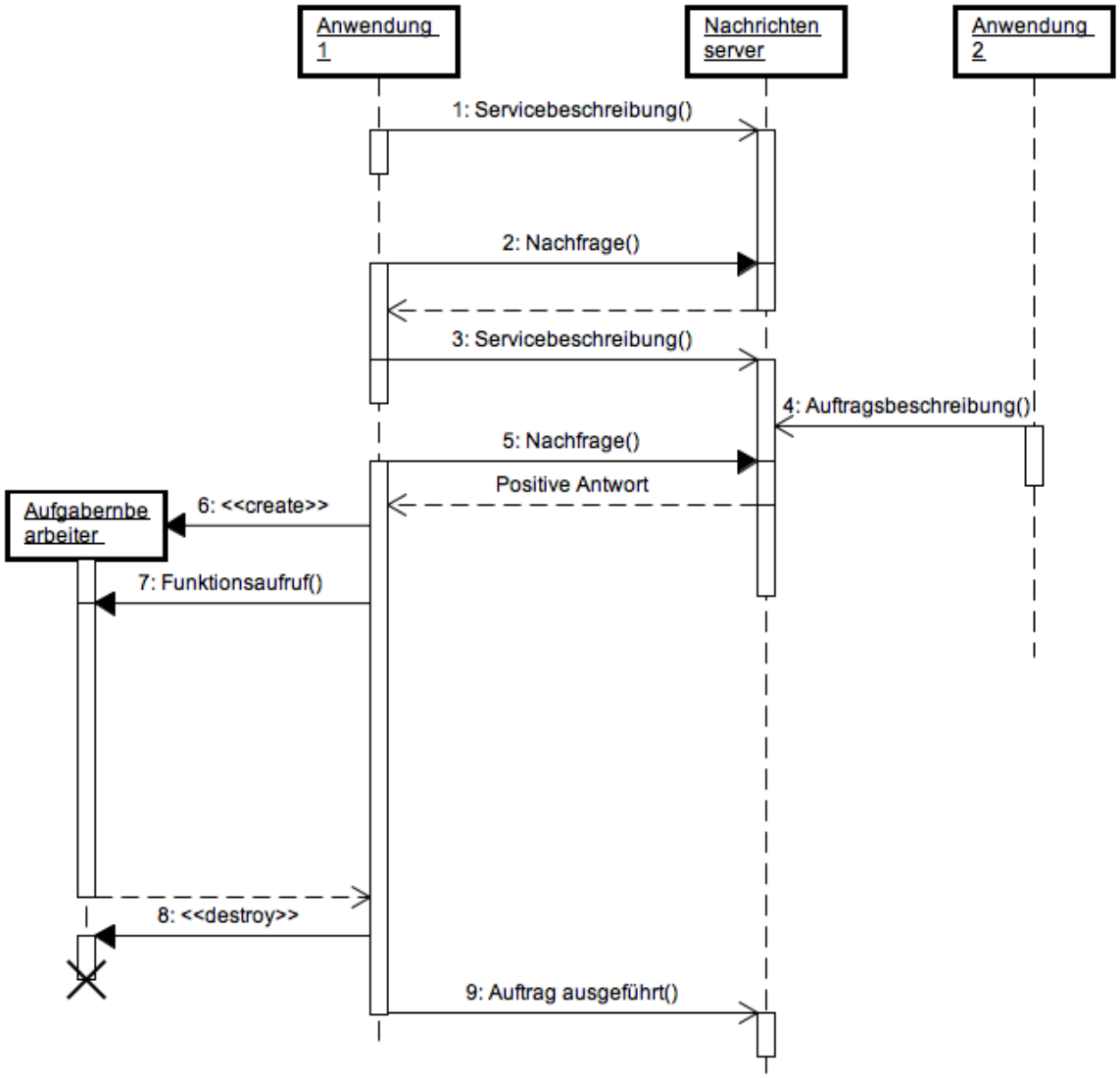

Abbildung 4.9.: Mit dem zentralen Nachrichtenserver verbinden sich beliebige Anwendungen

dem Observer-Pattern aus (Gamma u. a. 1994, 293-304). Eine weitere Möglichkeit ist es, nur eine Abfragemöglichkeit für hinterlegte Nachrichten anzubieten. Interessierte Parteien müssen dann regelmäßig anfragen. Dieser Polling-Mechanismus ist ein Kompromiss zwischen dem Ziel, Komponenten unabhängig zu halten und Effizienz im Regelfall.

Der Abonnementmechanismus hat den Nachteil, daß der Eventverteiler wissen muss, welche Anwendungen momentan verbunden sind und was ihre Interessen sind. Dies hat zur Folge, daß Anwendungen nicht entscheiden können, ob ein Mangel an Ereignissen[5] durch einen Absturz des Servers verursacht wurde. Nach dem Neustart eines Servers sind Informationen über Interessen verloren und müssen erst wiedergewonnen werden.

Wenn die Nachrichten gepollt werden, sind beim Server keine Informationen über Clients notwendig, da diese aktiv Informationen abfragen. Die Clients haben auch keine Schwierigkeiten zwischen einem Serverausfall und Mangel an Ereignissen zu unterscheiden.

4.4.2.1. Die Blackboardarchitektur — Eine eventbasierte Koordinationsstruktur

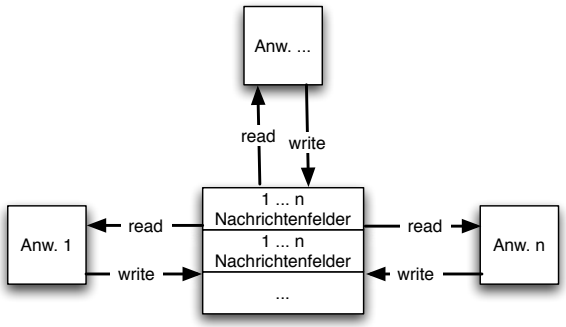

Abbildung 4.10.: In einer Blackboardarchitektur scharen sich eine Menge gleichberechtigter Anwendungen um einen gemeinsamen Datenraum und reagieren, wenn interessante Daten anliegen

Die Blackboardarchitektur ist eine Umsetzung einer datengetriebenen Aufrufstruktur, welche in (Erman u. a. 1980) beschrieben wurde. Hierbei gruppieren sich Prozesse um das gemeinsame Blackboard. Sie können auf dem Blackboard Nachrichten beliebigen Inhaltes ablegen und die vorhandenen Nachrichten abrufen. Wenn ihnen eine Nachricht bekannt vorkommt, führen sie eine Aktion aus, ohne daß der Nachrichtenschreiber hierüber eine Nachricht erhält. Bei dieser Kopplung von Blackboard und Prozessen müssen die einzelnen Prozesse

[5] Ab welcher Schwelle man auch einen solchen annimmt

keine Informationen übereinander haben. Javaspaces ist eine weitere Umsetzung dieser Idee, die Nachrichten wurden allerdings durch Javaobjekte ersetzt.

Diese Architektur nutzt den Ansatz, geteilte Speicherbereiche zur Auftragserteilung zu verwenden. Dabei werden Datenobjekte hinterlegt und von weiteren Prozessen interpretiert. Ein Blackboard erweitert und nutzt ihn jedoch zur allgemeinen Prozesskoordinierung. In (Johanson und Fox 2002) wurde der EventHeap beschrieben eine Javaimplementierung, die für den iRoom der Universität Stanford entwickelt wurde.

4.4.3. Fazit

Es wurden zwei Ansätze vorgestellt, die von Middlewares genutzt werden. Sie unterscheiden sich im Grad der Kopplung, die zwischen Objekten herrscht. Diese reicht von relativ stark[6] bis schwach[7]. Bei Nachrichtenbasierten Middlewares liegt der Grad der Kopplung zwischen diesen beiden Polen. Da die Abhängigkeit einzelner Komponenten gering sein soll, wird das System mit einer Middleware realisiert, welche datengetrieben agiert.

4.5. Schnittstellen

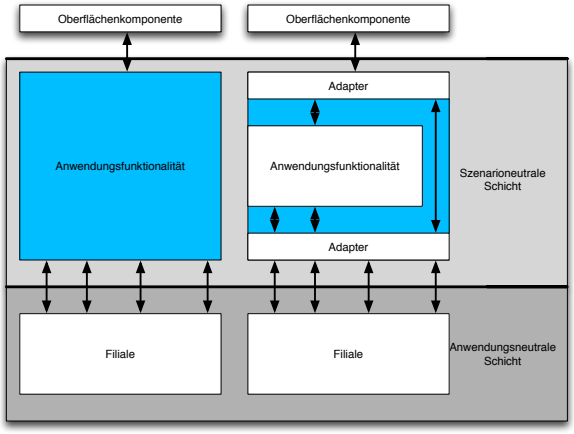

Abbildung 4.11.: Schnittstellen einer Applikation zur Benutzerschicht und zum System

Die einzelnen Komponenten bilden ein verteiltes System, das sich dem Benutzer allerdings als zusammenhängendes Ganzes darstellt. Um dies zu erreichen, werden die Komponenten

[6] Bei entfernten Methodenaufrufen.
[7] Bei datengetriebenen Systemen.

View, Controller, Modell und Persistenz über die Kommunikationskomponente verbunden. Zur Kooperation bieten die Komponenten Dienste an, mit denen ihre Funktionen genutzt werden können. So können Ausgabegeräte den Dienst "Darstellen" anbieten, über den die Views die Modelle ausgeben können.

In den folgenden Abschnitten werden die Schnittstellen zwischen Oberflächenkomponente und Anwendungsfunktionalität sowie zwischen Anwendungsfunktionalität und der Filiale beschrieben. An diesen Stellen sind festgelegte Schnittstellen notwendig, um die einzelnen Elemente sauber zu trennen und um die Koordinierung von Applikationen zu erleichtern, die keine Kenntnis voneinander haben.

4.5.1. Schnittstellen der Anwendungsfunktionalität

Diese Schnittstellen definieren das Verhalten einer Anwendung. Sie teilen sich auf in allgemeine Steuerungsschnittstellen und anwendungsspezifische Funktionen. In diesem Kapitel wird die Architektur des Gesamtsystems entworfen. Deshalb können in den folgenden Abschnitten nur die allgemeinen Schnittstellen definiert werden, die jede Anwendung unterstützen muss. Sollte eine Anwendung dies nicht tun, kann sie nicht oder nur eingeschränkt die Dienste des Systems nutzen.

4.5.1.1. Schnittstellen zur Oberflächenkomponente

Eine Sonderrolle spielen Funktionen die nur für die Oberflächenkomponente zu sehen sind. Diese haben zum Teil ebenfalls anwendungsunabhängige Funktionen, die jedoch in der Präsentation dem Szenario angepasst werden. Größtenteils wird hier jedoch die Anwendungsfunktionalität definiert. Die Anforderungen, die in dieser Schnittstelle abgedeckt werden, werden in (Neumann 2006) und (Burfeindt 2006) für Beispielszenarios beschrieben.

Folgende Bereiche werden hier abgedeckt:

Fenstermanagement Form, Größe und Position der Anwendungsfenster. Auch das Verschieben auf entfernte Displays wird hier angeboten.

Anwendungsfunktionalität Die anwendungsspezifischen Funktionen.

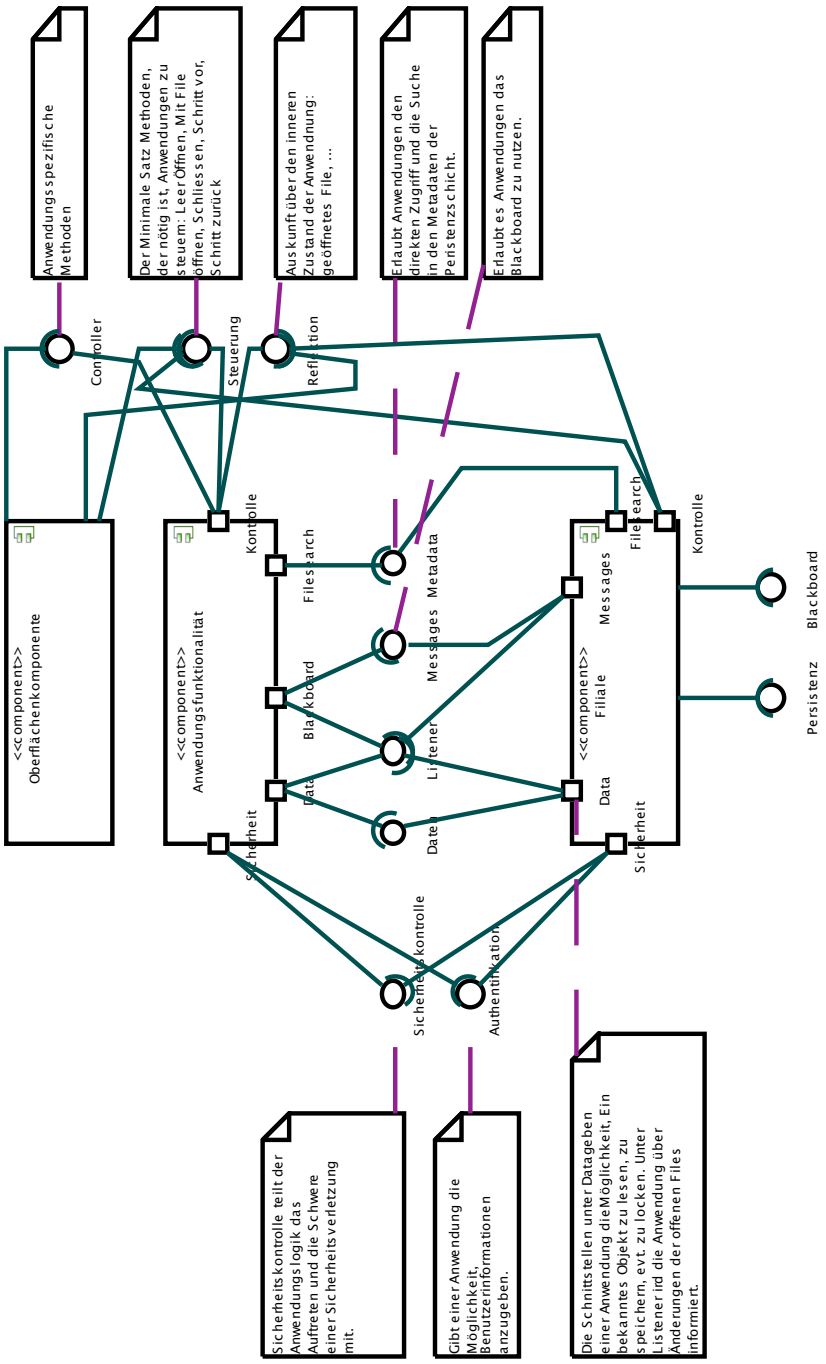

Abbildung 4.12.: Schnittstellen der Komponenten

4.5.1.2. Steuerung

Diese Schnittstelle ermöglicht es, jede Anwendung mit einer begrenzten Anzahl von Aktionen zu steuern. Mit ihrer Realisierung erfüllt eine Anwendung die in Abschnitt 2.4.5.2 gestellte Anforderung nach einem Satz anwendungsunabhängiger Operationen, die nötig sind, um beliebige Anwendungen zu koordinieren. In Abschnitt 2.4.5.2 wurde allerdings noch zwischen Operationen die Metadaten editieren und Funktionen zur Steuerung unterschieden. In diesem Abschnitt werden nur die Steuerungsoperationen definiert. Hierdurch ist es möglich, jede Art von Anwendung über eine generische Oberfläche zu kontrollieren. Die nächsten Funktionen kontrollieren das Behandeln von Dateien, vom Öffnen bis zum Schließen:

open() Öffnet ein neues Dokument

open(DocumentID id) Lädt und öffnet Dokument “id”

close() Schließt die grafische Ausgabe

save() Sichert den aktuellen Stand. Normalerweise sollte dies automatisch geschehen. Bei einigen Anwendungen kann es wünschenswert sein, dies manuell auszulösen.

Mit diesen Funktionen ist die Darstellung von Dokumenten auf entfernten Displays möglich.

Die folgenden Funktionen behandeln die Navigation in Dokumenten. Es ist allerdings die Sache des Anwendungsentwicklers, nicht-überraschende Interpretationen dieser Funktionen in seiner Anwendung zu implementieren. Eine mögliche Metapher ist eine Multimediafilebedienung, wie sie von CD-Spielern und DVD-Spielern bekannt ist. Eine weitere häufig genutzte Eingabemöglichkeit ist ein Pfeilkreuz. Die folgenden Methoden sind vorgesehen:

next(int Steps) Reaktion ist frei, z. B. `Steps` Seite weiter.

previous(int Steps) Reaktion ist frei, z. B. eine Seite zurück.

left(int Steps) Reaktion ist frei, z. B. Cursor `Steps` Schritte nach links.

right(int Steps) Reaktion ist frei, z. B. Cursor `Steps` Schritte nach rechts.

up(int Steps) Reaktion ist frei, z. B. Cursor `Steps` Zeilen nach oben.

down(int Steps) Reaktion ist frei, z. B. Cursor `Steps` Zeilen nach oben.

playForward() Starke Analogie zu einem Videospieler, Datei vorwärts abspielen.[8]

playBackward() Starke Analogie zu einem Videospieler, Datei rückwärts abspielen.[9]

stop() Aktuelle Aktion stoppen und an den Anfang des Files.

[8] Bei Audio und Vidoes natürliche Bedeutung.
[9] Bei Audio und Videos natürliche Bedeutung.

pause() Aktuelle Aktion stoppen und Position halten.

fastForward() Datei schnell abspielen.

fastBackward() Datei schnell und rückwärts abspielen.

seek(Position p) An Position p gehen.

Einige dieser Methoden scheinen nur Synonyme zu sein, so z. B. `next()`, `right()` und evt. `playForward()`. Dies ist nur bedingt so. So bezeichnet `next()` eine Bewegung in einer logischen Struktur, während `right()` eine Bewegung in einem zweidimensionalen Raum bezeichnet und `playForward()` den Start einer Bewegung im eindimensionalen Kontext eines Streams. (Dies können z. B. Audio– oder Videostreams sein). In speziellen Anwendungen können diese Befehle natürlich aufeinander abgebildet sein. Im Grunde ist dies das Design eines neuen Eingabegeräts wie z. B. einer Maus, bei dessen Entwurf man einen Kompromiss zwischen Handlebarkeit und Funktionalität eingehen muss.

4.5.1.3. Sicherheitskontrolle

Aus verschiedenen Gründen kann der Benutzer Aktionen auszulösen, zu deren Ausführung er keine Rechte hat. Weiterhin kann sich der Sicherheitsstatus von vorhandenen Dokumenten ändern, z. B. durch neu hinzugekommene Teilnehmer. Um der Anwendung darüber eine Rückmeldung zu geben und ihr zu ermöglichen, angemessen zu reagieren, existiert diese Schnittstelle. Sie muss alle Sicherheitsverletzungen, die im Sicherheitsmodell von (Mund 2006) möglich sind, behandeln können. Hierfür ist ein Eventmodell sinnvoll, da die Schnittstelle nicht angepasst werden muss, wenn neue Erkenntnisse oder Szenarioanforderungen die Art der möglichen Sicherheitsverletzungen verändern. Hierbei muss dann nur ein neues Sicherheitsevent definiert werden und sichergestellt sein, daß die Anwendung bei unbekannten Events eine Failsafe–Handlung definiert hat.[10] Ein erhöhtes Verständnis des Sicherheitsmodells durch die Anwendung äußert sich dann durch die Möglichkeit, weniger drastische Maßnahmen zu ergreifen.

Die Sicherheitsschnittstelle ist somit die Festlegung einer Ereignisbeschreibung. Um die Reaktionen mit einer kleinen Granularität anzupassen, ist jedoch ein Dokument nötig, das für Entwickler die zu beachtenden Ereignistypen und die verlangten Reaktionen beschreibt. Die Standardreaktion auf unbekannte Sicherheitsevents muss auf jeden Fall ein Verletzen der Sicherheitsvorschriften verhindern. Diese Reaktion ist vermutlich stärker als erforderlich und behindert die Arbeit mit dem System über Gebühr. Um dies zu vermeiden, muss die Eventhierarchie bekannt sein.

[10] So könnte ein Event die nötigen Schritte selber enthalten.

4.5.1.4. Reflektion

Um Informationen über den aktuellen Zustand einer entfernten Anwendung zu erhalten, wird dieses Interface implementiert. Sie ermöglicht es Informationen über angezeigte Dokumente und eine Bezeichnung der Anwendung zu erhalten. Dies ermöglicht eine Art "ps–Tool", oder ein Verzeichnis der im Netzwerk bearbeiteten Dokumente. Mit den Informationen, die über diese Schnittstelle verfügbar sind, ist die Möglichkeit gegeben, die in Abschnitt 2.4.5.2 formulierten Metadatenoperationen, auf den angezeigten Objekten auszuführen. Hierfür ist allerdings nicht mehr die Mitarbeit der befragten Anwendung nötig, dies kann direkt über die Schnittstelle zum Persistenzserver geschehen.

Mögliche Methoden dieser Schnittstelle wären:

getInfo() Liefert einen Statusbericht

getSecurityInfo() Liefert Informationen über bearbeitete Sicherheitsevents.

getName() Liefert den Namen, unter dem die Anwendung bekannt ist.

getCurrentDocumentID() Liefert die Objekt ID des aktuellen Dokuments.

In diesem Zustand hat diese Schnittstelle Probleme, mit zusammengesetzten oder mehreren angezeigten Objekten zurechtzukommen. Es wird angenommen, daß die Präsentationsschicht das aktuelle Dokument deutlich genug hervorhebt, um Missverständnisse zu vermeiden. Informationen zu Subobjekten müssen durch Navigation der Metadaten gesucht werden.

4.5.1.5. Zusammenfassung

Wenn eine Anwendungskomponente diese Schnittstellen unterstützt, kann sie im System eingesetzt werden. Diese Schnittstellen stellen allerdings nur ein Minimum dar. Eine Anwendung kann dem Benutzer einen deutlichen Mehrwert bieten, wenn auch die Schnittstellen der Filialkomponente verstanden und genutzt werden.

4.5.2. Schnittstellen der Filiale

4.5.2.1. Sicherheit

Wenn eine Anwendung noch nicht authentifiziert ist, besteht hier die Möglichkeit, sich am System anzumelden. Nach der Anmeldung ist der aktuelle Benutzer dem System bekannt.

Danach wird er bei Entscheidungen, ob Dokumente auf öffentlichen Displays gezeigt werden können, vom System berücksichtigt. Es sind zwei Methoden nötig: zum Anmelden und Abmelden.

4.5.2.2. Daten

Wenn einer Anwendung Dokumente zur Verfügung gestellt werden, so geschieht das über diese Schnittstelle. In der Filiale wird der Zustand der zentralen Kopie überwacht und die Anwendung bei Änderungen informiert. Sollten Änderungen entstanden sein, so werden diese über die Schnittstelle dem System bekanntgegeben. Eine solche Schnittstelle erlaubt es die verschiedenen Versionen eines Files zu erhalten. Auf sie kann wie auf ein lokales File zugegriffen werden, in kleinen Schritten und an beliebiger Position. Sie ermöglicht auch die Auskunft über die Größer der Daten.

4.5.2.3. Messages

Der Zugriff auf die zentrale Kommunikationskomponente, das Blackboard, findet über diese Schnittstelle statt. So muss eine Anwendung nicht die Adresse des Blackboards, oder die spezifischen Aufrufe kennen, um es zu nutzen. So könnte es eventuelldurch eine andere Lösung ersetzt werden, ohne die Anwendungen zu ändern. In dieser Schnittstelle können auch weitere Funktionen definiert sein, die der tatsächliche Server nicht bereitstellt.

writeMessage(Tuple) Schreibt eine Nachricht auf das Blackboard

listMessages(search parameter) Liest alle Nachrichten, die dem Suchparameter entsprechen

subscribeMessages(search parameter) Abonniert alle Nachrichten, die dem Suchparameter entsprechen

4.5.2.4. Metadaten

Dokumente sind nur über ihre Metadaten auffindbar, da die Speicherung nicht pfadbasierend erfolgt. Daher ist die Möglichkeit einer attributbasierten Suche erforderlich, um eine bestimmte Datei zu finden. Zusammen mit den eigentlichen Daten werden auch die Informationen über Erstellungszeit, Autor, Typ, Ort, etc. gespeichert. Diese Schnittstelle erlaubt es den Anwendungen, zu gewählten Parametern die passenden Objekt IDs zu bekommen.

4.5.3. Schnittstellen der Infrastruktur

Diese Interfaces werden von der Filiale genutzt, um die der Anwendungslogik angebotenen Funktionen zu realisieren. Für die Ablage von Nachrichten gibt es eine Kommunikationsschnittstelle auf niedriger Ebene. Zusätzlich existieren darüber laufende Protokolle, die das Suchen in der Metadatenbank erlauben.

4.5.3.1. Persistenz

Diese Schnittstelle wird vom gewählten Persistenzserver zur Verfügung gestellt und muss von der Filiale angesprochen werden. Der Zugriff auf den Persistenzserver wird durch ein Modul in der Filiale realisiert. Dieses muss beim Wechsel des Persistenzservers ausgetauscht werden. Das entsprechende Modul verarbeitet die Methoden der Schnittstellen `Daten` und `Metadaten`.

4.5.3.2. Blackboard

Diese Schnittstelle regelt den Nachrichtenaustausch mit der zentralen Komponente und anderen Applikationen. Diese Schnittstelle wird ebenfalls von der Infrastruktur zur Verfügung gestellt. Die Filiale muss daran angepasst werden. Man kann innerhalb der Filiale eine ähnliche Abstraktion wie für die Applikationen anstreben und die unterschiedlichen Schnittstellen in einem Modul der Filiale kapseln. Dies ist sinnvoll, da der Nachrichtenaustausch auch zu Realisierung weiterer Funktionen genutzt wird, für die das Blackboard nur die Kommunikationsschicht darstellt. Ein Beispiel hierfür ist die attributbasierte Objektsuche, die über das Blackboard abläuft, damit der Suchende nichts von der Existenz der Suchdienste wissen muss. Damit diese Suche funktioniert, muss eine Nachrichtenform existieren, die von beiden Parteien richtig interpretiert wird. Die Suche soll allerdings unabhängig von den Mechanismen der Nachrichtenveröffentlichung sein.

4.5.3.3. Zusammenfassung

Im Allgemeinen läuft die gesamte Kommunikation zwischen Filialen indirekt über das Blackboard. Dies geschieht mit dem Ziel, die Kommunikationspartner zu entkoppeln. Ein erwünschter Nebeneffekt ist allerdings, daß auch unbeteiligte Prozesse den Inhalt der Kommunikation mitlesen können und dadurch einen Überblick über die Ereignisse im Raum erlangen können.

4.6. Komponenten

Jede Komponente muss von den anderen unabhängig sein, so daß ein Ausfall einer Komponente nicht weitere Ausfälle im System nach sich zieht. Die Funktion zentraler Komponenten muss nach einem Ausfall schnell den alten Grad erreichen. Dies soll auch bei Übergabe der Verantwortung an ein neues Gerät funktionieren.

4.6.1. Kommunikation

Aus den Anforderungen ergeben sich verschiedene Gründe global zu kommunizieren. Dienste müssen veröffentlicht werden und nachfragbar sein. Damit Anwendungen zusammenarbeiten können, auch wenn sie vorher nicht gegenseitig bekannt waren, müssen Dienstanfragen, Ergebnisse und wichtige Statusänderungen der verschiedenen Applikationen öffentlich sein.

Ein Modell, das diese Möglichkeiten bietet, ist ein Blackboard, auf dem alle Nachrichten als Tupel veröffentlicht werden. Interessierte Anwendungen können nun den Inhalt des Blackboards mit gewissen Filtern abfragen.

Es wird nun ein zentraler Blackboardserver eingerichtet, auf dem alle Clients ihre Nachrichten veröffentlichen können. Die Struktur eines Blackboardsystems wurde in Abschnitt 4.4.2.1 beschrieben.

Durch diese Teilung der Kommunikation sind die Clients stark voneinander entkoppelt. Um zudem den Einfluss von Ausfällen weiter zu verringern, wird jede Nachricht mit einer bestimmten Gültigkeitsdauer versehen, so daß ausgefallene Dienste nach einer begrenzten Zeit nicht mehr im globalen Kontext auftauchen. Um diesen Effekt zu nutzen, dürfen Applikationen nicht auf die Erfüllung ihrer Anfragen oder auf die Antwort eines Dienstes warten.

4.6.1.1. Kommunikationsserver

Der Kommunikationsserver ist ein zentraler Rechner, der von allen Clients erreichbar sein muss. Von ihm wird ein Blackboard bereitgestellt, daß es erlaubt Nachrichten in Tupelform zu veröffentlichen. Da er die Semantik der Nachrichten nicht interpretiert, ist er auch für zukünftige Erweiterungen der Clientprotokolle offen.

Das Blackboard implementiert keine oder nur rudimentäre Sicherheitsfunktionen. Durch die freie Form der Nachrichten, hat der Server auch keine Möglichkeit, Entscheidungen zu treffen. Wenn solche Autorisationskontrollen gewünscht sind, muss der Blackboardclient geeignete Maßnahmen treffen.

4.6.1.2. Dienstverzeichnis

Um Dienste mit den gewünschten Funktionen bereit zu stellen, muss der Status der vorhandenen Dienste bekanntgegeben werden. Jede Dienstankündigung hat eine stark begrenzte Gültigkeitsdauer und muss danach wiederholt werden. Dies hält das Verzeichnis aktuell und vermeidet, daß Anwendungen auf abgestürzte Dienste warten. Abgestürzte Dienste werden dadurch schnell aus dem Verzeichnis entfernt und der aktuelle Zustand kann, nach einem Ausfall des Verzeichnisses, in einer bekannten Zeit vollständig wieder hergestellt werden. Daher muss die Verzeichniskomponente keine Maßnahmen treffen, um Informationsverluste zu vermeiden. Auch die Übernahme der Verantwortung ist hierdurch einfacher. Da die Ankündigungen nur kurz gültig sein sollen, eignet sich ein Blackboard für diese Rolle. Auf diesem können Nachrichten, mit festgelegter Struktur und kurzer Gültigkeitsdauer, die nötigen Informationen veröffentlichen. Während ein Dienst läuft, veröffentlicht er diese Nachrichten und bestätigt dadurch seine Erreichbarkeit.

Die nötigen Informationen sind:

- Art des Dienstes
- Kontaktinformationen
- Sicherheitseinschränkungen des Dienstes

4.6.1.3. Filiale

Die Clientseite des Blackboards läuft auf jedem Gerät und ermöglicht den dortigen Anwendungen Nachrichten zu veröffentlichen und zu lesen. Dies ist ein einfacher Mechanismus, um Anwendungen eines Typs zu koordinieren. Die Semantik der Nachrichten ist frei verhandelbar. Allerdings sollten einmal getroffene Entscheidungen dokumentiert werden, damit neue Anwendungen sie unterstützen können.

4.6.2. Applikation

Wie in Abschnitt 4.2 erwähnt, sind bei Anwendungen zwei Fälle zu unterscheiden. Einerseits an die Gegebenheiten des Raumes angepasste Programme, die die Anforderungen an synchrones Bearbeiten erfüllen sollen. Diese werden mit den Möglichkeiten, die das System bietet entwickelt. Andererseits vorhandene Geschäfts- oder Produktivitätsanwendungen, diese werden mit möglichst geringem Aufwand integriert.

Die Abbildung 4.11 legt bereits eine Trennung von Präsentation, Logik und Persistenz fest, während Abschnitt 4.5 die Schnittstellen zwischen diesen Bereichen vorgibt.

4.6.2.1. Neuentwickelte Anwendung

Die Anforderungen an neuentwickelte Anwendungen sind das Darstellen auf verschiedenen Endgeräten, Unterstützen mehrerer Benutzer, Synchronisieren mit anderen Anwendungen, synchrones und verteiltes Bearbeiten von Dokumenten. Diese Anforderungen sind auf mehrere Arten umsetzbar. Hier sollen Komponenten des Systems vorgestellt werden, die ein Erfüllen dieser Anforderungen unterstützen. Man kann Anwendungen konzeptionell in die verschiedene Komponenten aufteilen.

- Präsentation
- Anwendungsfunktionalität
- Daten

Um die Darstellung auf unterschiedlichen Geräten zu erlauben, die sich auch in ihren Fähigkeiten unterscheiden, werden Präsentation und Anwendungsfunktionalität nach dem Model–View–Controller–Muster aufgebaut. In diesem Muster werden die Elemente Model, View und Controller getrennt und nur noch durch wenige Methodenaufrufe verbunden. Ein– und Ausgabefunktionen (Controller und Views) sind unabhängig von der Anwendungsfunktionalität (den Modellen) und können leicht ausgetauscht, bzw. Anwendungsfunktionalität kann nun von mehreren View/Controller–Einheiten verwendet werden. Die Kommunikation könnte eventuellauch über die zentrale Kommunikationskomponente ablaufen. Das System unterstützt diese Trennung durch die Bereitstellung der Dienstinfrastruktur, sowie teilautomatisierte Erzeugung von Controllern.

Die Geräte können allerdings auch in ihren sonstigen Fähigkeiten voneinander abweichen. Da die Benutzer an verschiedenen Stellen im Raum sitzen und wiederum verschiedene Geräte zur Eingabe nutzen, müssen die Controller eines Objektes mit diesen verschiedenen Eingabequellen zurechtkommen.

Da Dienste über das öffentliche Blackboard kommunizieren, können betroffene Anwendungen wichtige Ereignisse mitbekommen und darauf reagieren. Damit dies auch für Veränderungen im View gilt, müssen auch diese eine Schnittstelle zum Blackboard haben. Hierdurch können mehrere Views durch einzelne Controller–Events gesteuert werden, ohne daß der Controller sie kennen muss.

Anwendungen können nun den Nachrichtenverkehr auf dem Blackboard lesen und auf interessante Nachrichten reagieren.

4.6.2.2. Oberflächenkomponente

View und Controller sind die Schnittstelle zum Benutzer und sollen die Möglichkeiten der Ein- und Ausgabegeräte optimal nutzen. Sie haben sowohl eine Verbindung zu den Modellen, als auch zu den Eventgeneratoren. Über diese werden die Benutzereingaben ins System eingebracht. Hierdurch wird eine Trennung der verschiedenen Aspekte der Eingabe, Ausgabe sowie Verarbeitung erreicht. Hiermit kann unter anderem die Forderung "mehrere Benutzer, ein View" und "mehrere Views, ein Benutzer" erfüllt werden.

4.6.2.3. Vorhandene Anwendungen

Bei vorhandenen Anwendungen ist die interne Struktur vermutlich weder einseh– noch veränderbar. Die Anwendung hat alle Funktionen, bis auf die Datenhaltung. Schnittstellen nach Außen können von extrem eingeschränkt bis zu komplett steuerbar reichen. Danach, und nach dem Grad der gewünschten Integration, richtet sich der Umfang der Anpassung.

Bei einer übernommenen Anwendung wird die Applikationslogik von der vorhandenen Applikationslogik übernommen. Um sie in die vorhandene Struktur einzubinden, muss ein Adapter geschrieben werden, der die Anwendung an die Schnittstellen anpasst. Je nach Umfang des Adapters und der umgesetzten Schnittstellenfunktionen, reicht der Grad der Integration von "Anzeigen und beenden", "Anzeigen und Steuern" bis "Öffnen und Synchron bearbeiten."

Beim ersten Fall reicht es aus, wenn der Wrapper das Dokument über das Filesystem zur Verfügung stellen kann. Der Rest der Interaktion funktioniert dann über Betriebssystemfunktionen. Bei Beenden der lokalen Anwendung wird das gespeicherte Dokument vom Wrapper zurück zum Persistenzserver gereicht.

In den letzten beiden Fällen muss die Anwendung eine umfangreiche API haben, damit der Wrapper die Funktionen als Dienst zur Verfügung stellen kann. Zusätzlich müsste hier allerdings noch ein neuer Controller geschrieben werden, der die Benutzeraktionen in Blackboardevents umsetzt.

4.6.2.4. Fazit

Sobald es mehrere Applikationen gibt, die auf denselben Daten mehrere Funktionen ausführen wollen, muss man das Modell erweitern, auch wenn die Funktionen keine große Verwandtschaft haben. Man könnte dies durch eine Trennung zwischen Daten– und Applikationsmodell erreichen, wobei das Datenmodell synchrone Zugriffe verwaltet. Dadurch wird auch ein großer Teil des Synchronisationsaufwands von den Applikationsprogrammierern

genommen. Dieses Problem ist bereits ausführlich im Zusammenhang mit Datenbanken untersucht worden. Zwei verbreitete Techniken zur Synchronisierung von parallelen Prozessen sind Semaphoren und Transaktionen.

4.6.3. Persistenzserver

Dieser Service ermöglicht den ortsunabhängigen Zugriff auf Dokumente, ihre verschiedenen Versionen und ihre Metadaten. Soweit möglich sollten diese auch im Dokument enthalten sein, dies wird jedoch nicht von allen Dokumententypen unterstützt. Bei lesendem wie schreibendem Zugriff, wird die Autorisation des Benutzers überprüft. Es gibt eine Schnittstelle nach Außen, durch die berechtigte Benutzer jederzeit auf die Daten zugreifen können. Es muss sichergestellt sein, daß das von (Mund 2006) ausgearbeitete Rechteschema nicht umgangen werden kann. Funktionen die vom Persistenzserver zur Verfügung gestellt werden umfassen:

- Suche nach Attributen
- Zugriff auf Daten / Metadaten
- Zugriff auf frühere / alternative Versionen

4.6.4. Benutzerdatenbank

Die Benutzerdatenbank wird als ein Service angeboten. Dieser erlaubt es den Komponenten des Systems, die Identität eines Benutzers festzustellen. Auch seine Gruppenzugehörigkeiten sind hier abgelegt. Mit diesen Informationen kann nun die Entscheidung getroffen werden, ob eine gewünschte Aktion ausgeführt werden darf.

4.6.5. Eingabegeräte

Eingabegeräte sind zunächst dem Gerät zugeordnet, mit dem sie hardwaremäßig verbunden sind, verbreiten allerdings ihre Verwendung über die Kommunikationskomponente. Hierdurch werden Eingabegeräte und verarbeitende Komponenten entkoppelt. Diese kommunizieren nur über die Kommunikationskomponente. Dies erlaubt es die eigenen Eingabegeräte (Maus, Keyboard, . . .) zur direkten Steuerung beliebiger Anwendungen zu nehmen.

Um diese Trennung umzusetzen, werden die Geräte als Eventgeneratoren betrachtet, die ihre Events auf dem Blackboard veröffentlichen. Zusätzlich beobachtet jeder Rechner, der entfernte Eingabegeräte nutzen möchte, das Blackboard. Wenn er Interaktionsevents sieht, die sein System betreffen nutzt er die veröffentlichten Events.

4.7. Entscheidungen

Die Kommunikationskomponente wird als ein Blackboard entworfen, da hierbei die Komponenten weniger Informationen brauchen. Andere Aufrufstrukturen, um spezielle Instanzen anzusprechen, können allerdings mit dem Blackboard als Kommunikationsmedium umgesetzt werden.

Bei den Anwendungen soll Präsentationsschicht und Anwendungsschicht durch die Verwendung des Observer–Musters verbunden bleiben, um die Darstellung flexibel zu halten, und eventuellspäter auch über das Netzwerk mit der Anwendungsschicht zu verbinden. Alle Aufgaben des Betriebssystem oder der Netzwerkdienste werden für die Anwendung von einer Bibliothek, der Filiale, gekapselt. Diese ist anwendungsneutral und kann für jede zu implementierende Anwendung verwendet werden.

Es gibt einen getrennten Persistenzserver, der eine direkte Schnittstelle zu den Filialen hat um die Nutzdaten der Objekte zur Verfügung zu stellen. Metadaten werden über Nachrichtenveröffentlichung auf dem Blackboard zur Verfügung gestellt.

4.8. Zusammenfassung

Es wurde die Architektur des Systems vorgestellt. Sie wurde in drei Schichten unterteilt, die den Bereichen Präsentation, Geschäftslogik und Persistenz entsprechen. Dies erfolgte, um einen Austausch unterschiedlicher Komponenten zu ermöglichen. Die Applikationslogik kann somit für die Realisierung unterschiedlicher Szenarios dienen, indem nur die Präsentationsschicht neu implementiert wird. Die Schichten werden über das Model / View / Controller–Musters aus (Gamma u. a. 1994) verbunden. Ebenso werden zwischen den Ebenen eine Reihe von Schnittstellen definiert, die die Kommunikation regeln. So wurde die Möglichkeit geschaffen Elemente der einzelnen Ebenen durch neue Elemente zu ersetzen. Dies kann für die Verbesserung der Effizienz nötig sein, oder um die Benutzerschnittstelle an andere Anforderungen anzupassen, wie in Kapitel 2 in Bezug auf heterogene Hardware gefordert wird. Auch die Anforderung nach dynamischer Konfiguration des Gesamtsystems wird durch die Verwendung einer Blackboardarchitektur erfüllt. Denn Geräte können, sofern sie die Kommunikationsschnittstellen implementieren, sofort im System aufgenommen werden. Ihre Anwesenheit muss dabei nicht explizit bekanntgegeben werden. Im Extremfall bleibt der Kommunikationsserver die einzige Komponente, die die Anwesenheit bemerkt. Trotzdem kann der Ablauf des Meetings Teilweise durch mitlesen der Blackboardkommunikation verfolgt werden.

5. Realisierung

In diesem Kapitel wird die Umsetzung der, im Designkapitel (4) ausgearbeiteten, Infrastruktur beschrieben. Hierfür werden soweit möglich Standardkomponenten eingesetzt. Darauf aufbauend wird gezeigt, wie eine einfache Beispielanwendung erstellt wurde.

5.1. Intention und Umfang

Die Infrastruktur wurde durch Realisierung der drei Komponenten soweit realisiert, so daß man die Tragfähigkeit des Konzeptes zeigen kann. Auf Infrastrukturkomponenten wie einen Sicherheitsserver wurde in diesem Schritt verzichtet, die Schnittstellen gegenüber den Anwendungen sind als Stubs ausgeführt worden.

Es wurde eine Präsentationssoftware "Polylux" geschrieben, welche die Funktionen der Filiale verwendet. Polylux soll die Anzeige unterschiedlicher Files auf mehreren Displays koordinieren. Durch die Nutzung der Middlewarefunktionen als Basis der eigenen Funktionalität, wird die Funktion der Middleware demonstriert. Hierbei erfolgt die Kommunikation verteilter Komponenten über die in Abschnitt 4.5 vorgestellte API. Die verschiedenen Dokumente werden durch die Filiale von einem zentralen Persistenzserver geladen, nachdem die Viewerkomponenten von Polylux sie anfordern. Dies zeigt die Tauglichkeit der Middleware und der Filiale, als Basis weiterer Anwendungen. Einige Komponenten wurden nur als Prototypen realisiert, die später durch die eigentlichen Implementationen ersetzt werden.

5.2. Laboraufbau

Das System wurde in einer beispielhaften Laborumgebung realisiert, die im Folgenden beschrieben wird. Sie weist einige Unterschiede zur späteren Vollausstattung aus. So gibt es nur einen Gerätetyp als Clientgerät und die öffentlichen Displays haben keine direkten Interaktionsmöglichkeiten. Es werden insgesamt fünf Rechner konfiguriert die unterschiedliche Rollen übernehmen.

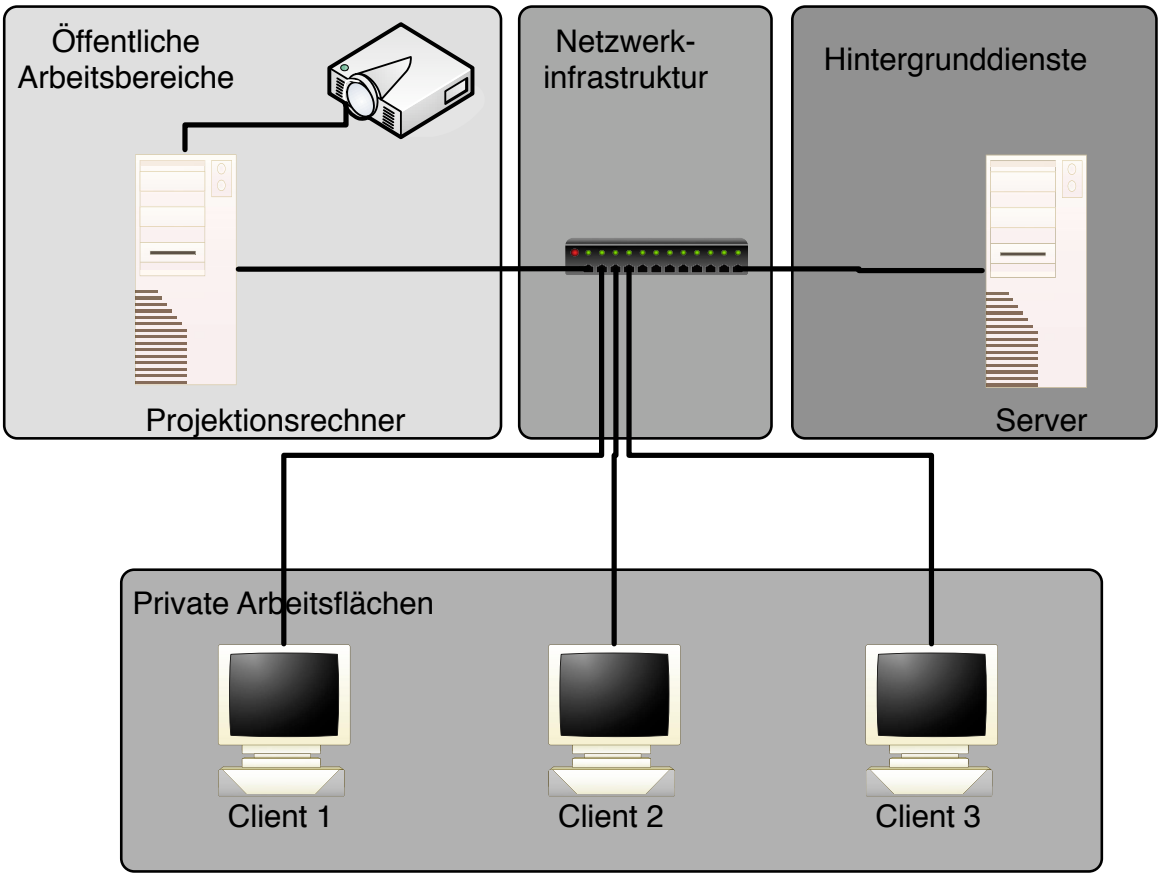

Abbildung 5.1.: Dieses Bild zeigt die verwendeten Komponenten, zum Aufbau der Hardwareumgebung des Prototyps.

5.2.1. Serverrechner

Als Serverrechner wurde ein normaler Desktop-PC verwendet. Das verwendete Betriebssystem ist Windows XP. Auf diesem Rechner laufen der Kommunikationsserver (Blackboardserver) und der Persistenzserver, die von den Filialen angesprochen werden können.

5.2.2. Projektorrechner

Als öffentliches Display wird ein an einen Videoprojektor angeschlossener Desktop-PC mit Windows XP verwendet. Dieser ist mit einer WLAN-Karte ausgestattet. Daher muss direkt am Aufstellungsort nur ein Stromanschluss vorhanden sein. Auf dem PC laufen mehrere Anwendungen, die die Nutzung des Projektors über das Netzwerk ermöglichen.

5.2.3. Clientrechner

Als Clientrechner kommen drei Computer mit installiertem Windows XP zum Einsatz. Auf ihnen werden die Komponenten der Anwendung installiert, die für die private Nutzung des Systems nötig sind.

5.2.4. Netzinfrastruktur

Die verwendete Netzwerkhardware besteht aus einem handelsüblichen DSL-Router mit angeschlossenem Switch und WLAN-Accesspoint. Im Prototyp wurde auf eine über die Nutzung von WEP hinausgehende Sicherung des Netzes verzichtet.

5.3. Programmiersprache

Als Sprache der selbst entwickelten Komponenten wird Java gewählt. Diese Sprache gewährleistet einerseits die Plattformunabhängigkeit des kompilierten Codes, die eine Hardwareunabhängigkeit der Clientkomponenten erleichtert. hinzu kommt, daß die gewählte Middleware als Javaimplementierung vorliegt und bei Wahl einer anderen Entwicklungsplattform zuerst die vorhanden Bibliotheken integriert, oder die vorausgesetzten Interfaces neu implementiert werden müssen.

5.4. Implementierung der Middleware

Um die Ziele aus Abschnitt 5.1 zu erreichen, mussten verschiedene Komponenten implementiert werden. Dazu zählen:

Filiale Die Filiale bietet den Anwendungen die Verbindung zu Diensten im Netz.

Blackboardserver Bietet den Filialen und dem Persistenzserver die Möglichkeit der Kommunikation.

Persistenzserver Bietet der Filiale die Möglichkeit durch Objekt IDs auf identifizierte Objekte zuzugreifen. Die Metadatensuche wurde in diesem Prototyp nicht implementiert.

Auf Basis dieser drei Komponenten wurden die Anwendungen "Fernsteuerung" und "Polylux" implementiert. In den folgenden Abschnitten wird zuerst die Realisierung der Infrastruktur beschrieben, danach, in den Abschnitten 5.5.1 und 5.5.2, die Beispielanwendungen.

5.4.1. Blackboardserver

Der Blackboardserver dient als Kommunikationsserver. Er steht den Filialen als Blackboard zur Verfügung. Beim suchen nach vorhandenen Implementationen stieß ich auf Jini Javaspaces, und den iROS EventHeap. Jini nutzt Javatechnologien, um dem Client eine protokollunabhängigen Zugriff zu ermöglichen. Der EventHeap nutzt ein in (Johanson und Salgar 2002) beschriebenes Protokoll, das man in mehreren Programmiersprachen implementieren könnte. Um für später entwickelte Clients nicht auf eine bestimmte Sprache angewiesen zu sein habe ich den EventHeap als Blackboardserver eingesetzt. Dieser braucht nach dem Starten keine Konfiguration und ist somit sofort einsatzbereit.

5.4.2. Filiale

Bei iROS ist eine Javabibliothek zum Zugriff auf Dienste im Netz enthalten. Ihre Schnittstellen entsprechen jedoch nicht vollständig den in Kapitel 4.5 definierten, so daß für sie noch ein Adapter geschrieben werden muss. Diese Adapter ergeben dann zusammen die anwendungsneutrale Komponente. Einige der definierten Schnittstellen haben keine Entsprechung bei iROS diese müssen neu implementiert werden. Für den Prototyp wurden die folgenden Klassen entwickelt:

SecurityMonitor Überwacht die Aktionen der anderen Klassen und verhindert Rechteverletzungen.

BlackboardAccess Bietet der Anwendung Zugang zum Blackboard.

FileAccess Baut die Verbindung zum Persistenzserver auf übertragt die Daten.

Steuerung Liest die Nachrichten auf dem Blackboard und übersetzt sie in Steuerungsbefehle.

5.4.3. Persistenzserver

Die Funktion des Persistenzservers wurde in diesem Prototyp ein WebDAV Server verwendet. Dieser bietet einen auf Objekt IDs basierenden Filezugriff und wurde mit einem Subversionbackend konfiguriert, das einen Zugriff auf alte Versionen der gespeicherte Files erlaubt. Im Prototypen wurde die Funktionalität des Lesens und Schreibens, auf die aktuelle Version, implementiert. Die Beantwortung von Suchanfragen auf dem Blackboard ist nicht implementiert worden.

5.5. Anwendung

Um die Prinzipien des Raumes zu zeigen wird zuerst eine einfache Anwendung gezeigt, die das entfernte Öffnen von Files erlaubt. Danach werden die erstellten Komponenten erweitert, um die Funktionalität von Polylux zu erhalten.

5.5.1. Fernsteuerung

Dies ist die erste Anwendung, die auf dem System laufen soll. Der Benutzer kann aus einer Liste von verfügbaren Geräten auswählen und dort eine Anwendung mit einem vorgegebenen File starten. Die gestartete Anwendung kann er mit seiner lokalen Maus steuern, die Mausbewegungen werden von dem Programmpaar Eingabesender und Eingabeempfänger übertragen.

Abbildung 5.2 zeigt die Verteilung der Anwendungen auf die Rechner der Laborumgebung. Es werden zur Demonstration drei der Rechner genutzt. Der Server, der Projektorrechner und Client 1. Hierbei laufen auf dem Projektorrechner ein Eingabeempfänger und ein Metadaemon. Beide reagieren auf Anwenderereignisse, die auf das Blackboard geschrieben wurden. Zusätzlich ist ein Anwendungsstarter vorhanden, der vom Metadaemon angestoßen wird. Auf dem Clientrechner laufen ein Eingabesender, der es erlaubt, Eingabeereignisse auf das Blackboard zu schreiben und ein Fernstarter, der die Ansteuerung von Metadaemons erlaubt.

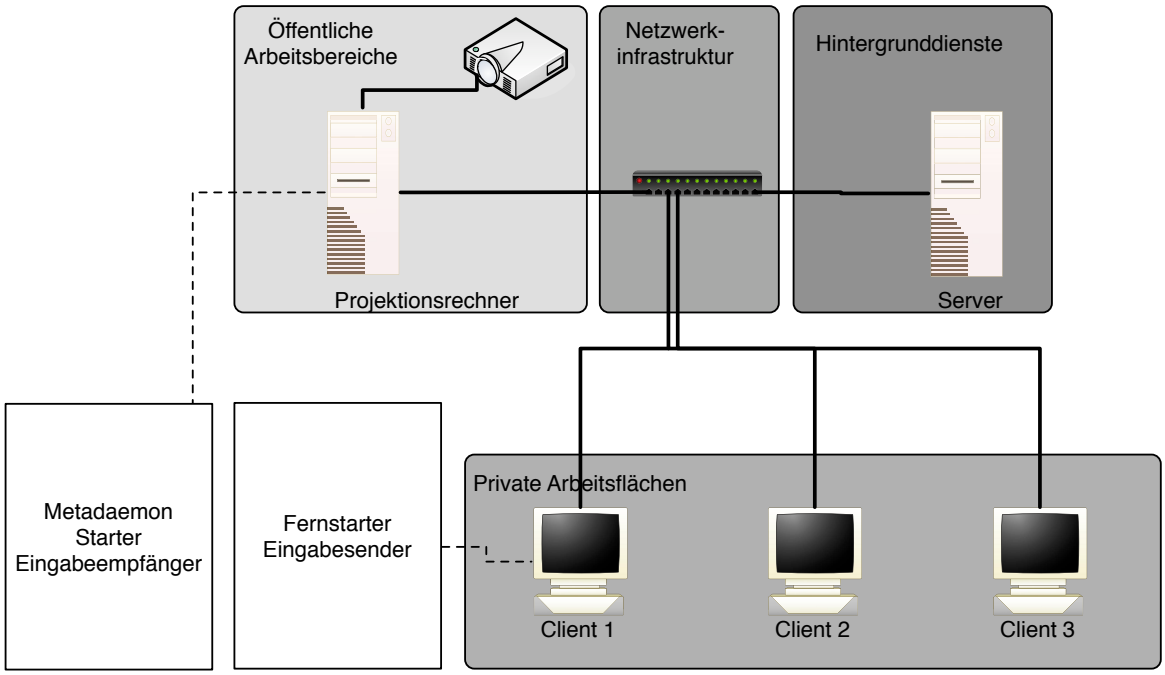

Abbildung 5.2.: Die Rollen der Geräte bei der Anwendung Fernsteuerung

Eingabesender Eine Komponente, die die Position einer angeschlossenen Maus, in einem virtuellen Raum, auf dem Blackboard hinterlegt.

Eingabeempfänger Eine Anwendung, die für einen Teil des virtuellen Raums, den der Maussender beschreibt, verantwortlich ist. Liest das Blackboard, ob gerade Zeigegeräte in ihrem Zuständigkeitsbereich sind. Diese werden dann in Eingabeereignisse der Zielplattform umgesetzt.

Fernstarter Bietet dem Anwender eine Schnittstelle zu den Metadaemonen auf dem Projektorrechner.

Metadaemon Stellt sich auf dem Blackboard vor und sucht nach Nachrichten für ihn. Wenn er eine erkennt, ruft er den gewünschten Dienst auf.

Anwendungsstarter Nimmt eine Objekt ID entgegen, untersucht die Metadaten, um den Objekttyp zu erkennen und startet eine Anwendung, die im Betriebssystem für diesen Typ registriert ist. Dieser Anwendung werden die Daten des Objekts zur Verfügung gestellt.

In den folgenden Abschnitten werden die einzelnen Komponenten weiter beschrieben.

5.5.1.1. Eingabesender

Die Funktionen sollten in diesem Prototyp von PointRight2 übernommen werden. (Johanson u. a. (2000), (Johanson u. a. 2002)) Dieses bietet die Möglichkeit entfernte Rechner mit den eigenen Eingabegeräten zu steuern. Der Eingabesender besteht aus einem Modul, das die Signale der lokalen Geräte mitliest, und in einen virtuellen Arbeitsbereich abbildet. In diesem hat auch der lokale Bildschirm eine Repräsentation. PointRight greift direkt auf die Schnittstellen des Blackboards zu und natürlich nicht die Filiale. Leider ist es mir nicht gelungen PointRight an unsere Topologie anzupassen. Aus beiden Gründen wurden die einfachen Entsprechungen Mausempfänger und Sender entwickelt. Diese haben das Konzept eines virtuellen Raumes, in dem sich die Eingabe bewegt.

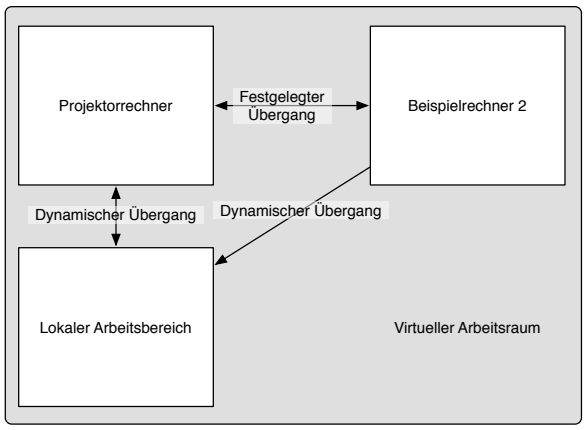

Abbildung 5.3.: Der Aufbau des virtuellen Arbeitsraumes

Aufbau des Virtuellen Arbeitsraums Der Virtuelle Arbeitsraum kann als Graph mit gerichteten Kanten beschrieben werden. Die einzelnen Knoten stehen hierbei für Flächen mit festgelegter Größe, die einem Ausgabegerät entsprechen.

Dieser Graph wird vom Eingabesender verwaltet. In dessen Konfigurationsdatei sind die ansprechbaren Displays festgelegt, sondern auch die möglichen Übergange. Die Navigation erfolgt durch die Interpretation des Graphen als endlicher Automat. Die Übergange zwischen den einzelnen Zuständen werden nun durch festgelegte Übergangsereignisse ausgelöst. Clients, deren Arbeitsflächen keine spezielle Entsprechung im Automaten haben, werden auf einen Standardanfangszustand abgebildet.

Solange der Eingabefokus innerhalb dieser Repräsentation ist, bleibt die Anwendung für den Anwender transparent. Sobald ein Arbeitsplatzrand erreicht wird, werden die weiteren Eingabeevents auf das Blackboard geschrieben. Hierbei wird der Name das aktuellen Displays

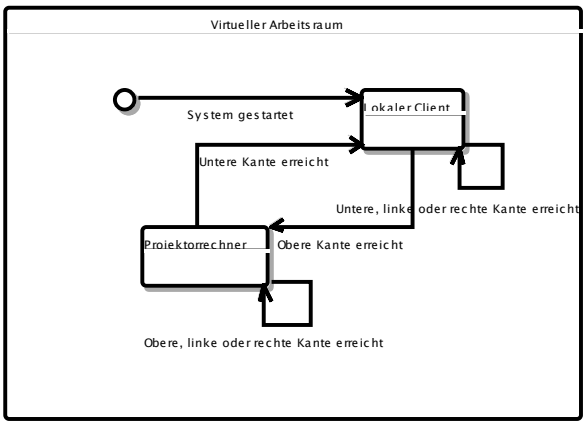

Abbildung 5.4.: Die Konfiguration des virtuellen Arbeitsraumes in der Laborumgebung, dargestellt als Automant

genannt und die absoluten Koordinaten genannt. Das aktuelle Display ergibt sich hierbei aus dem aktiven Zustand des Konfigurationsautomaten. Dem Benutzer wird hierbei eine geeignete Rückmeldung darüber gegeben, daß der lokale Bildschirm nicht mehr im Ziel der Eingaben liegt. Die Konfiguration entspricht der Darstellung in Abbildung 5.4

5.5.1.2. Eingabeempfänger

Der Eingabeempfänger besteht aus einer Komponente, die über die Filiale Nachrichten der Eingebesender auf dem Blackboard liest. Ihr wurde in der Konfiguration ein Name übergeben, der dem Knoten des virtuellen Arbeitsraums entspricht, für den sie verantwortlich ist. Wenn es auf dem Blackboard Nachrichten eines Eingabesenders gibt, die ihren Namen enthalten, werden die Eingabeevents dieser Nachricht gelesen und an das Betriebssystem weitergereicht. Weiterhin veröffentlicht der Eingabeempfänger seinen Namen und die Eigenschaften der von ihm betreuten Arbeitsfläche. Dies ermöglicht es späteren Versionen des Eingabesenders, seine Konfiguration dynamisch an die Situation im Netzwerk anzupassen.

5.5.1.3. Anwendungsstarterdaemon

Der Anwendungsstarter veröffentlicht den Namen seines Rechners, zusammen mit seiner Dienstbezeichnung, auf dem Blackboard und kann dadurch gezielt von Clients angesprochen werden. Er reagiert auf Nachrichten, die seinen Namen, seine Dienstbezeichnung und

eine Objekt ID enthalten, indem er eine Anwendungsstarterinstanz erzeugt und ihr die Objekt ID übergibt.

5.5.1.4. Anwendungsstarter

Der Anwendungsstarter ermittelt den Typ der Daten und bestimmt die geeignete Anwendung[1]. Der Anwendung werden die Daten zur Verfügung gestellt und auf Änderungen überwacht. Sollten Änderungen auftreten, werden die Daten auf den Filestore geschrieben. Sollte die Filiale mitteilen, daß sich die Daten auf dem Persistenzserver geändert haben, wird die lokale Anwendung neu geöffnet, oder aktualisiert.

Der Zustand der lokalen Anwendung wird überwacht. Wenn die lokale Anwendung beendet werden sollte, beendet sich auch der Anwendungsstarter, unter der Annahme, daß das Prozessende vom Benutzer herbeigeführt wurde. Beim Beenden wird die temporäre Datei auf Änderungen untersucht und, nach eventuellem Speichern, auf dem Persistenzserver gelöscht.

In späteren Funktionen ist es denkbar, der lokalen Anwendung einen Filestream oder ähnliches zur Verfügung zu stellen, um die Menge an übertragenen Daten zu minimieren. In diesem Prototyp wird die Datenübergabe durch eine temporäre Datei realisiert.

5.5.1.5. Fernstarter

Der Fernstarter durchsucht das Blackboard nach Ankündigungen von Metadaemons. Wenn er Ankündigungen gefunden hat stellt er sie mit Name und eventuellvorhandenen Informationen dem Benutzer dar. Der Benutzer hat nun die Möglichkeit eine Objekt ID mit den dargestellten Metadaemons zu verbinden. Der Fernstarter erstellt daraufhin eine Nachricht, die den Namen des Metadaemons, die Dienstbezeichnung und die Objekt ID enthält.

5.5.1.6. Fazit

Die vorgestellten Elemente ermöglichen es, Anwendungen auf entfernten Rechnern zu starten. Diesen Anwendungen muss ein im Netzwerk vorhandenes Objekt übergeben werden. Danach kann die Anwendung durch die Verwendung von Eingabesender und Eingabeempfänger benutzt werden.

[1] Durch Betriebssystemfunktionen (Filetype unter Windows) oder eine eigene Liste

5.5.2. Polylux

Als Demonstration soll eine verteilte Präsentationsanwendung geschrieben werden, die es ermöglicht mehrere Dokumente in mehreren Formaten koordiniert wiederzugeben. Sie soll aus zwei Verwaltungskomponenten bestehen, die das Benutzerinterface zur Verfügung stellen sowie einer Sammlung von Darstellungsdiensten, die über das Netz initialisiert und gesteuert werden.

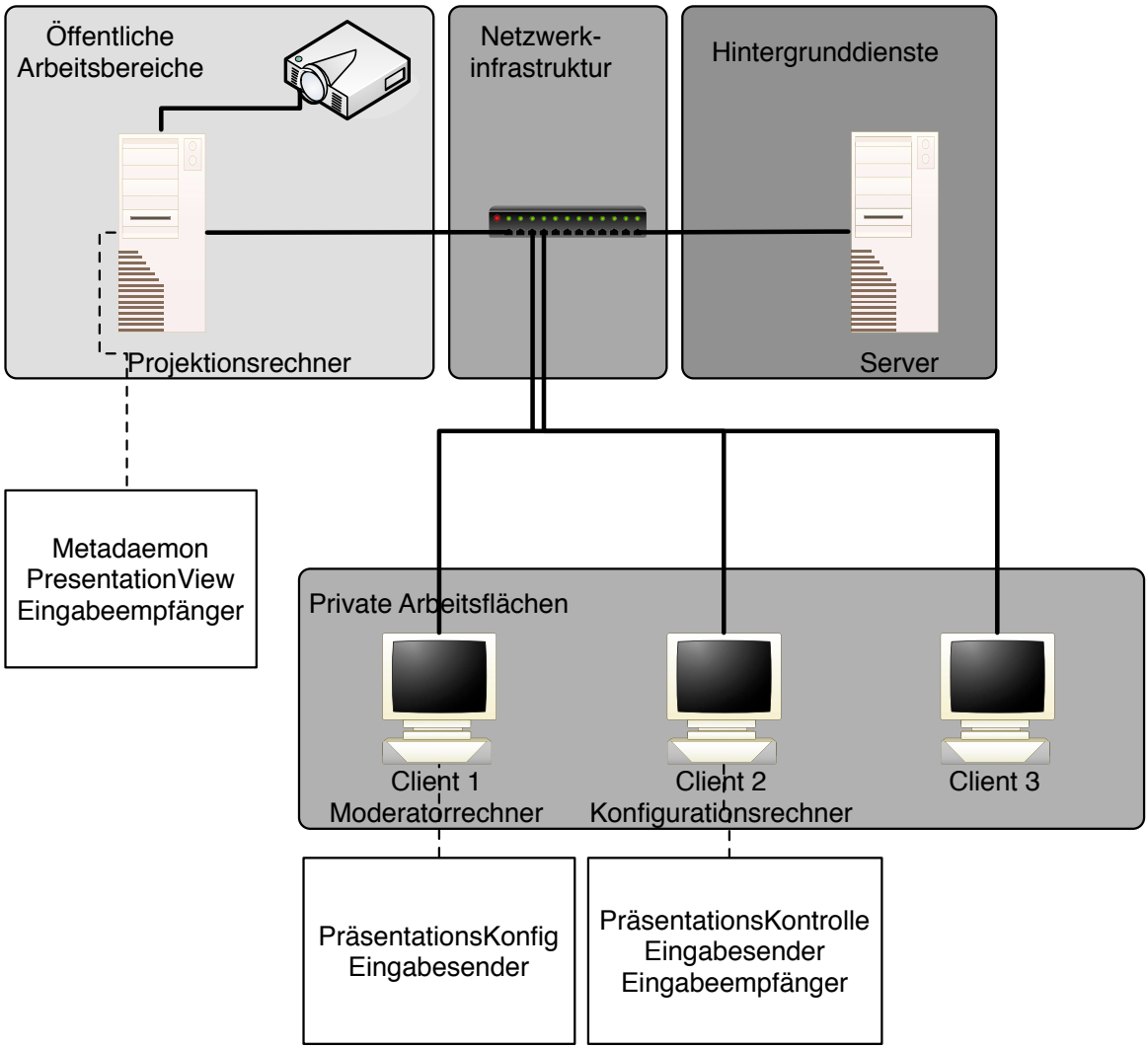

Abbildung 5.5.: Die Rollen der Geräte bei der Anwendung Polylux

Die nötigen Komponenten sind:

Präsentationskonfigurator Erweiterter Fernstarter

Präsentationskontrolle Erlaubt die Kontrolle mehrerer Präsentationsviewer

Präsentationsviewer Stellt ein File dar und reagiert auf die Signale der Präsentationskontrolle.

5.5.2.1. Präsentationskonfigurator

Der Präsentationskonfigurator stellt im Netzwerk laufende Präsentationsviewer und ihren Status dar. Der Benutzer kann sie nun mit einer Objekt ID verknüpfen. Beim Starten der Präsentation, sendet er den konfigurierten Viewern die gewählte Objekt ID sowie den Namen der Präsentation.

5.5.2.2. Präsentationskontrolle

Die Präsentationskontrolle legt Steuernachrichten auf dem EventHeap ab, diese beinhalten die gewünschte Aktion aus der Schnittstelle Steuerung und den Namen der Präsentation. Der Präsentationsname kann vom Benutzer eingegeben werden.

5.5.2.3. Präsentationsviewer

Die Anzeigekomponenten werden auf den entfernten Rechnern gestartet und stellen das assoziierte Dokument dar. Bei Erzeugung werden ihnen Filterbedingungen für das Blackboard mitgegeben, mit denen sie nach Steuersignalen suchen. Die Viewer bestehen hierbei aus vorhandenen Anwendungen, die von einem Javawrapper die Befehle zum Starten und Fortschreiten im Dokument erhalten. Hierdurch sind sie nicht mehr so flexibel, in der Art der angenommen Objekte, wie die Anwendungsstarter, da die Adapter jetzt mehr Funktionen der einzelnen Anwendungen ansprechen müssen. Dies bedeutet, das man für jeden Dokumententyp einen eigenen Präsentationsviewer braucht der den Dokumententyp interpretieren kann.

5.6. Fazit

Die Implementierung zeigte, daß ein die gewählte Architektur mit einem Blackboard sich gut für eine Koordination unterschiedlicher Anwendungen eignet. Dies bestätigt die Erfahrungen der "Interactive Workspaces" Gruppe Stanfords, die diese Architektur für ihren iRoom wählten. Bei der Installation der weiteren Komponenten von iROS mussten viele absolute Pfade angepasst werden. Da der Persistenzserver und eine Sicherheitskomponente ohnehin noch

hinzukommen sollen, wurde auf den Einsatz der Beispielanwendungen verzichtet, auch um den Ersatz an die festgelegten Schnittstellen anzupassen.

Bei der Entwicklung von Polylux wurden die zur Verfügung gestellten Funktionen der Filiale genutzt, um eine verteilte Anwendung zu schaffen. Ein zusätzlicher Navigator könnte einen Überblick über angezeigte Dokumente und aktive Steuerungskomponenten bieten. Dies bietet die Möglichkeit, einen Teil der der Präsentation auch auf einem lokalen Display anzuzeigen.

Insgesamt hat sich gezeigt, daß der Entwurf zur Umsetzung von verteilten Arbeiten geeignet ist. Auch wenn nur ein Teil der Komponenten realisiert ist, bietet ein solcherart ausgestatteter Konferenzraum einen Mehrwert gegenüber der vorherigen Version.

6. Fazit und Ausblick

Die Vision des Ubiquitous Computing ist durch die Fortschritte in der Hardwareentwicklung einen Schritt näher gerückt, viele Menschen tragen elektronische Geräte ständig mit sich. Gleichzeitig ist sie jedoch immer noch in weiter Ferne, da sich die Art der Interaktion durch den Computer nicht grundlegend verändert hat.

Die Dringlichkeit zur Realisierung der Vision wird allerdings höher, denn in den letzten Jahren hat sich dem einzelnen Menschen eine immer größere Zahl von elektronischen Informationskanälen erschlossen. Dies begann im 19. Jahrhundert mit der Erfindung des Telefons und setzte sich über Fernsehen, Fax, Mobiltelefon, Email und dem Internet fort. Die Informationskanäle erfordern momentan einen nicht unerheblichen Teil unserer Aufmerksamkeit. Wenn die Menschen sich wieder Neuem widmen sollen, muss es Werkzeuge geben, die diesen Strom beherrschbar machen, statt einen weiteren Teil der Aufmerksamkeit zu fordern.

Groupwareanwendungen sind auf eine kritische Menge an Benutzern angewiesen, daher müssen sie intuitiv zu erfassen und allgegenwärtig verfügbar sein. So lässt sich die Menge potentieller Anwender erhöhen. Das langfristige Ziel ist das Verschwinden des Computers in den Hintergrund und die Konzentration auf die eigentlichen Aufgaben der Anwender (Weiser 1991). Durch die Verfügbarkeit ubiquitärer Hintergrunddienste wird die Möglichkeit geschaffen, fachspezifische CSCW-Anwendungen zu entwickeln.

Bei Meetings nehmen sich alle Teilnehmer Zeit, um gemeinsame Aufgaben zu lösen. Hierfür müssen sie auf eine Menge von vorhandenen Informationen zurückgreifen. Dadurch wirken Verbesserungen in Konferenzräumen besonders stark, da gleich mehrere Teilnehmer betroffen sind. Die vorliegende Arbeit versucht innerhalb des Konferenzraums eine Allgegenwärtigkeit der angebotenen Funktionen zu erreichen.

Die Arbeit zeigt die Möglichkeiten auf, die ein Konferenzraum bieten kann, wenn die vorhandenen Geräte durch geeignete Mittel verbunden werden. Hierfür wurden die Anforderungen an die Gruppenarbeit in einem Konferenzraum erarbeitet. Außerdem wurde anhand eines Beispielszenarios erläutert, wie Arbeit effektiver gesteltet werden könnte.

Aus der Szenariobeschreibung und den fachlichen Anforderungen ergeben sich eine Reihe technischer Anforderungen, wobei sich die vorliegende Arbeit unter anderem an existierenden Projekten orientiert hat (Tandler (2004), Johanson (2003), Department of Computer Science (2005), MIT (2004)). Die Anforderungen beschreiben ein offenes System, das durch

neue Hardware oder Software erweitert werden kann und die Integration eines Sicherheitsmonitors vorsieht. Erarbeitete Ergebnisse sollen in allen Versionen, ortsunabhängig und ortstransparent, verfügbar sein. Der Zugriff auf Dokumente wird durch ein Sicherheitsmodell geregelt, dadurch können auch vertrauliche Dokumente ohne Bedenken bearbeitet werden. Durch die offene Architektur kann das System für Anwendungen in verschiedenen Fachdomänen genutzt werden.

Die Untersuchung existierender Produkte aus den Bereichen CSCW und Groupware hat gezeigt, daß einzelne Produkte nicht die Anforderungen an ein intuitives Interface erfüllen. Sie stellen in der Regel ihre Funktionen in den Vordergrund, nicht die Aufgabe des Benutzers. Die Forschungsprojekte aus den Bereichen Ubiquitous Computing und Mensch-Computer-Interaktion erfüllten die Anforderungen zu einem großen Teil. Sie hatten jedoch die Sicherheitsaspekte ausgespart ((Johanson 2003, 186-187), (Tandler 2004, 202)).

Nach dieser Untersuchung wurde eine Architektur entworfen, die es früh ermöglicht, nutzbare Anwendungen zu entwickeln. Diese Architektur lässt sich später durch Einsetzen von neuen Komponenten an weitere Anforderungen anpassen. Sie enthält eine Abstraktionskomponente (Filiale) auf Clientseite, einigen Schnittstellen zu den Clients und einen zentralen Server zum Nachrichtenaustausch. Die Funktionen des Persistenzservers und einer Sicherheitskomponente sind bereits vorgesehen, das System kann jedoch auch ohne sie genutzt werden.

Diese Architektur wurde dann durch die Verwendung von vorhandenen Komponenten für den Kommunikationsserver und den Persistenzserver umgesetzt. Durch die Entwicklung zweier Beispielanwendungen wurde gezeigt, welchen Mehrwert eine Vernetzung bieten kann.

Bei der Umsetzung der Filiale wurden die Schnittstellen zur Nachrichtenverwaltung, zum Datenteil des Persistenzservers und zur Steuerung realisiert. Andere Schnittstellen sind nur als Stubs ausgeführt worden, da die Komponenten der Middleware, die angesprochen werden sollten, noch nicht implementiert waren. Dies sind die Sicherheitskomponente und die Metadatenkomponente des Persistenzservers.

Als wichtigste Aufgabe bleibt die Ergänzung des Systems mit den Komponenten Persistenzserver und Sicherheitsmonitor. Diese werden von Horst Mund konzipiert und realisiert. Danach ist die Architektur in ihrer Grundfunktionalität umgesetzt. Es ist denkbar, eine Unterstützung für verteilte Objekte zu integrieren. Dies ermöglicht dann ein paralleles Arbeiten nicht nur auf den Metadaten, sondern auch direkt auf den Dokumenten.

Die Erarbeitung und Umsetzung der Anforderungen anderer Fachdomänen können den Raum für eine breitere Nutzerschicht attraktiv machen. Ein Vergleich der verschiedenen Anforderungsprofile kann die Erweiterung eines fachdomänenunabhängigen Werkzeugkastens ermöglichen.

Um die Suchfunktion des entwickelten Systems zu einem mächtigeren Werkzeug zu machen, kann man untersuchen, wie die Werkzeuge des Semantic Web zu integrieren sind. Auch die Ergebnisse aus Abschnitt 2.4.5.1 könnten weiter vertieft werden, in dem ein paralleles Mindmappingtool realisiert und seine Benutzung untersucht wird.

Der realisierte Raum schafft eine ubiquitäre Umgebung, in der neue Werkzeuge der Zusammenarbeit möglich sind. Die ursprüngliche Vision ist ein Ideal, welches nie erreicht werden kann, da jede neue Funktion den Blick auf neue Möglichkeiten eröffnet.

Daher sollte man beobachten, wie der so ausgestattete Konferenzraum genutzt wird und welche Verhaltensweisen auftreten. In diesem Zusammenhang wäre es interessant zu wissen, ob die von (Winograd und Flores 1987) und (Weiser 1991) erwünschte Dominanz der Aufgaben ermöglicht wird. Aus dem beobachteten Benutzerverhalten lassen sich auf jeden Fall Rückschlüsse auf weitere Werkzeuge ziehen. Eine weitere Integration der physischen Objekte ist ebenfalls denkbar (Ringel u. a. 2003).

Das Feld der CSCW ergibt zusammen mit dem Paradigma des Ubiquitous Computing ein weites Feld an Möglichkeiten. Als paralleles Forschungsgebiet sollte man die Nutzung eines solchen Raumes zur Unterstützung verteilter Zusammenarbeit untersuchen. In einer globalen Gesellschaft wird hierfür großer Bedarf bestehen.

Literaturverzeichnis

Beck und Andres 2004 BECK, Kent ; ANDRES, Cynthia: *Extreme Programming Explained: Embracing Change*. 2. Addison-Wesley, 2004 15

Berners-Lee u. a. 1998 BERNERS-LEE, T. ; FIELDING, R. ; IRVINE, U.C. ; MASINTER, L.: Uniform Resource Identifiers (URI): Generic Syntax / IETF — Network Working Group. URL `http://www.ietf.org/rfc/rfc2396.txt`, 1998. – Request for Comments 105

Borriell u. a. 2005 BORRIELL, Gaetano ; CHALMERS, Matthew ; LAMARCA, Anthony ; NIXON, Paddy: Delivering REAL-WORLD Ubiquitous Location Systems. In: *Communications of the ACM* 48 (2005), Nr. 3, S. 36–41

Brumitt u. a. 2000 BRUMITT, Barry ; MEYERS, Brian ; KRUMM, John ; KERN, Amanda ; SHAFER, Steven A.: EasyLiving: Technologies for Intelligent Environments. In: *HUC*, URL `http://research.microsoft.com/easyliving/Documents/2000HUC.pdf`, 2000, S. 12–29 28

Burfeindt 2006 BURFEINDT, Lars: *Diplomarbeit HAW Hamburg 2006, in Vorbereitung*, HAW Hamburg, Diplomarbeit, 2006 13, 15, 20, 27, 60

Caronni und Waldvogel 2003 CARONNI, Germano ; WALDVOGEL, Marcel: Establishing Trust in Distributed Storage Providers. In: *Proceedings of the Third International Conference on Peer-to-Peer Computing (P2P?03)*, IEEE, 2003

Chiu u. a. 1999 CHIU, Patrick ; KAPUSKAR, Ashutosh ; WILCOX, Lynn ; REITMEIER, Sarah: Meeting Capture in a Media Enriched Conference Room. In: *Proceedings of CoBuild'99* FX Palo Alto Laboratory and University of Michigan School of Information (Veranst.), URL `http://www.fxpal.com/publications/FXPAL-PR-99-118.pdf`, 1999

Department of Computer Science 2005 DEPARTMENT OF COMPUTER SCIENCE, University Of Illinois at Urbana-Champaign: *Active Spaces for Ubiquitous Computing*. Website. 2005. – URL `http://gaia.cs.uiuc.edu/` 85

Ellis u. a. 1991 ELLIS, Clarence A. ; GIBBS, Simon J. ; REIN, Gail: Groupware: some issues and experiences. In: *Commun. ACM* 34 (1991), Nr. 1, S. 39–58. – ISSN 0001-0782 10

Erman u. a. 1980 ERMAN, Lee D. ; HAYES-ROTH, Frederick ; LESSER, Victor R. ; REDDY, D. R.: The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. In: *ACM Computing Surveys* 12 (1980), June, Nr. 2, S. 213–253 58, 103

Gamma u. a. 1994 GAMMA, Erich ; HELM, Richard ; JOHNSON, Ralph ; VLISSIDES, John M.: *Design Patterns: Elements of Reusable Object-Oriented Software*. 23. print, 2002, ISBN: 0-201-63361-2. Reading, MA, USA : Addison-Wesley, Oktober 1994. – ISBN 0-201-63361-2 54, 58, 72

Grossmann u. a. 2001 GROSSMANN, Matthias ; LEONHARDI, Alexander ; MITSCHANG, Bernhard ; ROTHERMEL, Kurt: A World Model for Location-Aware Systems. In: *Informatik — Zeitschrift der schweizerischen Informatikorganisationen* 5 (2001), Nr. 15, S. 23–26. – URL `http://www.svifsi.ch/revue/pages/issues/n015/in015.pdf` 28

Guimbretière u. a. 2001 GUIMBRETIÈRE, François ; STONE, Maureen ; WINOGRAD, Terry: Fluid Interaction with High-resolution Wall-size Displays / Computer Science Department, Stanford University. 2001. – Forschungsbericht 11

Intille u. a. 2003 INTILLE, Stephen S. ; LEE, Vivienne ; PINHANEZ, Claudio: Ubiquitous Computing in the Living Room: Concept Sketches and an Implementation of a Persistent User Interface / Massachusetts Institute of Technology and IBM T.J. Watson Research. 2003. – Forschungsbericht

Janneck JANNECK, Michael: *Fachbereich Mensch-Computer-Interaktion: Ziele und Aufgaben*. Website. – URL `http://www.mensch-computer-interaktion.de/ziele-aufgaben.html` 104

Johanson u. a. 2000 JOHANSON, B. ; HUTCHINS, G. ; WINOGRAD, T.: PointRight: A System for Pointer/Keyboard Redirection Among Multiple Displays and Machines / Stanford University. URL `http://graphics.stanford.edu/papers/pointright/`, March 2000. – Forschungsbericht. 2000, Stanford, CA: Stanford University 79

Johanson 2003 JOHANSON, Brad: *Application Coordination Infrastructure for Ubiquitous Computing Rooms*, Stanford University, Dissertation, Dezember 2003. – URL `http://graphics.stanford.edu/~bjohanso/dissertation/` 85, 86

Johanson und Fox 2002 JOHANSON, Brad ; FOX, Armando: The Event Heap: A Coordination Infrastructure for Interactive Workspaces / Stanford University. 2002. – Forschungsbericht 40, 42, 43, 59

Johanson u. a. 2002 JOHANSON, Brad ; HUTCHINS, Greg ; WINOGRAD, Terry ; STONE, Maureen: PointRight: Experience with Flexible Input Redirection in Interactive Workspaces. In: *Proceedings of UIST-2002*. Gates 3B-376, Serra Mall. Stanford, CA

94305-9035, USA 191 Pine Lane, 2002. – URL http://graphics.stanford.edu/papers/pointright-uist2002/ 79

Johanson u. a. 2001 JOHANSON, Brad ; PONNEKANTI, Shankar ; KICIMAN, Emre ; SENGUPTA, Caesar ; FOX, Armando: *System Support for Interactive Workspaces*. 03 2001. – URL http://graphics.stanford.edu/papers/iwork-sosp18/ 10, 42, 43, 45

Johanson und Salgar 2002 JOHANSON, Brad ; SALGAR, Satyajeet: The Wire Protocol for the Event Heap (as of Event Heap 1.96) / Stanford University. URL http://iwork.stanford.edu/docs/eheap/wire_protocol.html, März 2002. – Website 76

Johnson-Lenz und Johnson-Lenz 1994 JOHNSON-LENZ, Peter ; JOHNSON-LENZ, Trudy: *Groupware: Coining and Defining It*. Website. April 1994. – URL http://www.awakentech.com 10

Kahlbrandt 2001 KAHLBRANDT, Bernd: *Software-Engineering mit der Unified Modeling Language*. 2. Springer Verlag, 2001 51

Knop 2004 KNOP, Michael: *UbiMex: Ein persönliches Wissensmanagementsystem für Ubiquitous Computing*, HAW Hamburg, Diplomarbeit, 2004. – URL http://download.ubimex.org/thesis.pdf 18, 28

Lahlou u. a. 2005 LAHLOU, Saadi ; LANGHEINRICH, Marc ; RÖCKER, Carsten: Privacy and Trust Issues with Invisible Computers. In: *Communications of the ACM* 48 (2005), Nr. 3, S. 59

Microsoft 2006 MICROSOFT: *LiveMeeting*. Website. 2 2006. – URL http://office.microsoft.com/de-de/FX010909711031.aspx 34

MIT 2004 MIT: *MIT Oxygen Overview*. Website. 2004. – URL http://www.oxygen.lcs.mit.edu/Overview.html 85

Mund 2006 MUND, Horst: *Diplomarbeit HAW Hamburg 2006, in Vorbereitung*, HAW Hamburg, Diplomarbeit, 2006 21, 26, 28, 63, 71

Netviewer GmbH NETVIEWER GMBH: *Netviewer Tools*. Website. – URL http://www.netviewer.de/netviewer-produkte.html 34

Neumann 2006 NEUMANN, Carola: *Effizienzsteigerung konzeptioneller Diskussions- und Entscheidungsprozesse, Diplomarbeit HAW Hamburg 2006, in Vorbereitung*, HAW Hamburg, Diplomarbeit, 2006 13, 15, 20, 27, 60

Prante u. a. 2002 PRANTE, Thorsten ; MAGERKURTH, Carsten ; STREITZ, Norbert: Developing CSCW Tools for Idea Finding - Empirical Results and Implications for Design. In: *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW 2002)*, ACM Press, 2002, S. Noch nicht eingetragen 7, 12, 25, 42

Raasch 1992 RAASCH, Jörg: *Systementwicklung mit Strukturierten Methoden*. 2. Hanser, 1992. – ISBN 3-446-17263-7 51

Richter u. a. 2001 RICHTER, Heather ; ABOWD, Gregory D. ; GEYER, Werner ; FUCHS, Ludwin ; DAIJAVAD, Shahrokh ; ; POLTROCK, Steven: Integrating Meeting Capture within a Collaborative Team Environment. In: *Proceedings of the International Conference on Ubiquitous Computing, Ubicomp 2001*, Georgia Institute of Technology and IBM T.J. Watson Research Center and Boeing Mathematics & Computing Technology, 2001, S. pp 123–138.. – URL `http://home.cc.gatech.edu/hrichter/uploads/20/Richter-Ubicomp2001.pdf` 32

Ringel u. a. 2004 RINGEL, Meredith ; RYALL, Kathy ; SHEN, Chia ; FORLINES, Clifton ; VERNIER, Frederic: Release, Relocate, Reorient, Resize: Fluid Techniques for Document Sharing on Multi-User Interactive Tables. In: *CHI 2004*, 2004

Ringel u. a. 2003 RINGEL, Meredith ; TYLER, Joshua ; STONE, Maureen ; BALLAGAS, Rafael ; BORCHERS, Jan: iStuff: A Scalable Architecture for Lightweight, Wireless Devices for Ubicomp User Interfaces / Stanford University Department of Computer Science. 2003. – Forschungsbericht 87

Russell u. a. 2002a RUSSELL, Daniel M. ; DREWS, Clemens ; SUE, Alison: Social Aspects of Using Large Public Interactive Displays for Collaboration / IBM Almaden Research Center. 2002. – Forschungsbericht 7, 16, 35, 36, 37

Russell u. a. 2005 RUSSELL, Daniel M. ; STREITZ, Norbert A. ; WINOGRAD, Terry: Building disappearing Computers. In: *Communications of the ACM* 48 (2005), Nr. 3, S. 42–48 12, 40

Russell u. a. 2002b RUSSELL, Daniel M. ; TRIMBLE, Jay ; WALES, Roxana: Two paths from the same place: Task driven and humancentered evolution of a group information surface / IBM Almaden and NASA. 2002. – Forschungsbericht 36, 37

Schlichter u. a. 2001 SCHLICHTER, Johann ; REICHWALD, Ralf ; KOCH, Michael ; MÖSLEIN, Kathrin: Rechnergestützte Gruppenarbeit (CSCW). In: *i-com Zeitschrift für interaktive und kooperative Medien* (2001), S. 5–11. – URL `http://sunschlichter13.informatik.tu-muenchen.de/servlet/Item_Query?cmd=listitems&mode=detail&itemid=34` 103

Scholtz u. a. 2003 SCHOLTZ, Jean ; ARNSTEIN, Larry ; KIM, Miryung ; KINDBERG, Tim ; CONSOLVO, Sunny: User-centered Evaluations of Ubicomp Applications / Intel Corporation. 2003 (IRS-TR-02-006). – Forschungsbericht 13, 32

Schümmer u. a. 2000 SCHÜMMER, Jan ; SCHÜMMER, Till ; SCHUCKMANN, Christian: COAST – Ein Anwendungsframework für synchrone Groupware. In: *Conference Proceedings for the "Net.ObjectDays 2000"*, URL `http://www.opencoast.org/`, 2000 41, 103

Shih u. a. 2004 SHIH, Clara C. ; FOX, Armando ; WINOGRAD, Terry ; SZYBALSKI, Andy ; CRONÉ, Maria: Teamspace: A Simple, Low-Cost and Self-Sufficient Workspace for Small-Group Collaborative Computing / Computer Science Department Stanford and Department of Computer and Systems Science Stockholm University/Royal Institute of Technology. 2004. – Forschungsbericht 40, 43

Smart Technologies SMART TECHNOLOGIES: *Bridgit*. Website. – URL `http://www2.smarttech.com/st/en-US/Products/Bridgit/default.htm` 34

Stanford University, HCI Group 2005 STANFORD UNIVERSITY, HCI GROUP: *Human Computer Interaction*. Website. 2005. – URL `http://hci.stanford.edu/research/ideas/` 42

Streitz und Nixon 2005 STREITZ, Norbert ; NIXON, Paddy: The disappearing Computer. In: *Communications of the ACM* 48 (2005), Nr. 3, S. 33–35 9

Streitz u. a. 1999 STREITZ, Norbert A. ; GEISSLER, Jörg ; HOLMER, Torsten ; KONOMI, Shin ichi ; MÜLLER-TOMFELDE, Christian ; REISCHL, Wolfgang ; REXROTH, Petra ; SEITZ, Peter ; STEINMETZ, Ralf: i-LAND: An interactive Landscape for Creativity and Innovation. In: *Proceedings of the ACM Conference on Human Factors in Computing Systems*, ACM Press, April 1999, S. pp. 120–127. 104

Tandler u. a. 2002 TANDLER, P. ; STREITZ, N. ; PRANTE, T.: *Roomware – Moving Toward Ubiquitous Computers*. 2002. – URL `citeseer.ifi.unizh.ch/tandler02roomware.html` 11, 41

Tandler 2003 TANDLER, Peter: The BEACH Application Model and Software Framework for Synchronous Collaboration in Ubiquitous Computing Environments. In: *JSS - Ubi Tools* (2003) 20

Tandler 2004 TANDLER, Peter: *Synchronous Collaboration in Ubiquitous Computing Environments — Conceptual Model and Software Infrastructure for Roomware Components*, Technische Universität Darmstadt, Dissertation, 2004 10, 41, 45, 85, 86

Tanenbaum und van Steen 2002 TANENBAUM, Andrew S. ; STEEN, Martin van ; APT, Alan (Hrsg.): *Distributed Systems: Principles and Paradigms*. 1. Upper Saddle River; New Jersey 07458 : Prentice Hall, 2002. – URL `http://www.prenhall.com/tanenbaum` 50

Wagner u. a. 2006 WAGNER, Dominik ; OTT, Martin ; PITTENAUER, Martin: *SubEthaEdit*. Website. 01 2006. – URL `http://www.codingmonkeys.de/subethaedit/` 38

Weiser 1991 WEISER, Mark: The Computer for the 21st Century. In: *Scientific American* 265 (1991), September, Nr. (3), S. 94?104. – URL `http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html` 10, 85, 87, 104

Winograd und Flores 1987 WINOGRAD, Terry ; FLORES, Fernando: *Understanding Computers and Cognition: A New Foundation for Design (Language and Being)*. Addison-Wesley, 1987. – URL `http://hci.stanford.edu/winograd/` 10, 26, 87

A. Anwendungsfalltabellen

Anwendungsfall	**Systemadministration**
Zielbeschreibung	Die Objekte des Systems werden angelegt und ihre Relationen zueinander festgelegt.
Precondition	Wissen um die gewünschten Änderungen
Success End Condition	Die interne Modellierung stimmt wieder mit den abgebildeten Strukturen überein.
Primary Actor	Administratoren
Trigger	Der Zustand des Systems hat sich geändert/soll sich ändern. z. B. ein neuer Benutzer, neue Geräte, neue Gruppen
Description	Step | Action 1 | Am Gerät anmelden 2 | Aktion auswählen 3 | Aktion bearbeiten 4 | Abmelden
Sub Variations	Step | Branching Action 2 | Benutzer:Anlegen 2 | Benutzer:Ändern 2 | Benutzer:Löschen 2 | Benutzer:Gruppenzugehörigkeit ändern 2 | Geräte:LIPDs anmelden 2 | Geräte:Gruppieren 2 | Geräte:Gerätedaten ändern

Tabelle A.1.: Anwendungsfall: Systemadministration

Anwendungsfall	**Benutzeraccount anlegen**
Zielbeschreibung	Ein neuer Benutzer wird dem System bekannt gemacht.
Precondition	Es gibt ein neues Teammitglied, das integriert werden muss.
Success End Condition	Das neue Teammitglied kann das System nutzen.
Failure End condition	Das Teammitglied ist von der Arbeit am System ausgeschlossen.
Primary Actor	Administrator
Trigger	Team erhält ein neues Mitglied
Description	Step: 1 — Action: Voraussetzungen für Benutzeraufnahme prüfen 2 — Benutzerdaten erfassen 3 — Plausibilität der Daten prüfen 4 — Account erstellen 5 — Erfolg bestätigen und Accountdaten mitteilen
Extensions	Step: 4a — Branching Action: Teammitglied ist extern : <action or name of sub.use case>

Tabelle A.2.: Anwendungsfall: Benutzer anlegen

Anwendungsfall	**Benutzeraccount löschen**
Zielbeschreibung	Wenn ein Mitglied das Team verlässt, soll sein Zugang zum System gesperrt werden.
Precondition	Wir wissen, welches Mitglied die Firma verlässt.>
Success End Condition	Ehemaliges Mitglied, kann System nicht mehr nutzen.
Primary Actor	Administrator

Tabelle A.3.: Anwendungsfall: Benutzeraccount löschen

Anwendungsfall	**Dienste Nutzen**
Zielbeschreibung	Benutzer arbeiten zusammen und bearbeiten dabei Objekte und ihre Metadaten
Primary Actor	1 Benutzer
Description	Step | Action 1 | Am Gerät anmelden 2 | Dienst nutzen 3 | Vom Gerät abmelden
Sub Variations	Step | Branching Action 2 | Objektverwaltung 2 | Abrufen 2 | Anzeigen 2 | Einfügen 2 | Bearbeiten

Tabelle A.4.: Anwendungsfall: Dienste nutzen

Anwendungsfall	**Ein Dokument aus dem Repository holen**
Zielbeschreibung	Dokumente sind im System gespeichert. Um sie zu nutzen, müssen sie zuerst in den Arbeitskontext des Benutzers geholt werden.
Precondition	Benutzer kennt Eigenschaften des gesuchten Objektes
Success End Condition	Im hat, in seinem Arbeitsbereich, Zugriff auf das Objekt
Failure End condition	Der Benutzer hat kein Objekt gefunden, das seinen Ansprüchen entsprach.
Primary Actor	Benutzer
Trigger	Benutzer braucht Zugriff auf ein Objekt.
Description	Step: 1 — Action: Benutzer legt Suchkriterien fest. 2 — Ergebnismenge wird geliefert 3 — Auswahl der Visualisierung 4 — Auswählen des Objektes / der Objekte
Sub Variations	Step: 1 — Branching Action: Nach Autoren, Teilnehmern, Änderungsdatum, anderen Metadaten, Inhalt 3 — Zeitlinie, Autoren, Team, Projekt, . . .

Tabelle A.5.: Anwendungsfall: Abrufen von Objekten

Anwendungsfall	**Dokument im System einfügen**
Zielbeschreibung	Bevor Dokumente genutzt werden, müssen sie zuerst im System eingefügt werden. Dies kann ein neuangelegtes Dokument sein, welches im System mit Inhalt erfüllt wird, oder ein bestehendes Dokument, welches weitergeschrieben werden kann.
Success End Condition	Ein neues mit Metadaten versehenes Objekt ist im Repository
Failure End condition	Repository im alten Zustand
Primary Actor	Benutzer
Description	Step / Action 1 / Art festlegen 2 / Name festlegen 3 / Evt. Weitere Metadaten 4 / Objekt wird unter Einbeziehung von Autor, sowie Ort und Zeit der Erstellung abgelegt

Tabelle A.6.: Anwendungsfall: Dokument einfügen

Anwendungsfall	**Objekt bearbeiten**
Zielbeschreibung	Objekte die vorher abgerufen oder eingefügt wurden, können bearbeitet werden. Änderungen werden an allen Instanzen gezeigt.
Precondition	Ein Objekt wurde abgerufen, es sind Schreibrechte vorhanden.
Success End Condition	Die Änderungen sind erfolgreich, der Benutzer ist als Autor dieser Änderungen festgehalten, Aktivitätsanzeige des Objektes ist aktualisiert.
Failure End condition	Änderung wurde verworfen. Änderung wurde aufgeschoben.
Primary Actor	Benutzer
Description	Step: 1 — Action: Methode auswählen 2 — Parameter übergeben 3 — Parameter an Objekt senden 4 — Objekt gibt Änderung bekannt. 5 — Anzeigen passen sich an.
Extensions	Step: 4a — Branching Action: Änderung kollidiert mit anderer Benutzeraktion :Konfliktauflösung
Sub Variations	Step: 1 — Branching Action: Methodenwahl kann explizit oder implizit erfolgen. 2 — Parameter werden durch Stift, Maus, Tastatur, ... übergeben

Tabelle A.7.: Anwendungsfall: Objekt bearbeiten

Anwendungsfall	**Objekte anzeigen**
Zielbeschreibung	Objekt auf einer Anzeige darstellen
Precondition	Objekt wurde aus dem Repository geholt
Success End Condition	Ein Objekt wird auf einem Display angezeigt
Failure End condition	Das Objekt ist nicht sichtbar
Primary Actor	Benutzer
Trigger	Benutzereingabe

Tabelle A.8.: Anwendungsfall: Objekt anzeigen

Anwendungsfall	**Objekte verwalten**
Zielbeschreibung	Ändern der Metainformationen von Objekten und Benutzern; Löschen von Objekten
Primary Actor	Benutzer; Admin
Sub Variations	Branching Action Löschen Projektzugehörigkeit Gruppenzugehörigkeit Attribute Setzen

Tabelle A.9.: Anwendungsfall: Objektverwaltung

Anwendungsfall	**Objekte Löschen**
Zielbeschreibung	Manchmal ist es notwendig, Objekte endgültig zu löschen. Fragen:Endgültig? Aus der Teamkontinuität? Aus den Verbindlichkeiten?
Success End Condition	Das Objekt ist nicht mehr im System vorhanden
Failure End condition	Nichts hat sich verändert.
Primary Actor	Benutzer

Tabelle A.10.: Anwendungsfall: Objekt löschen

Anwendungsfall	**Projektzugehörigkeit ändern**
Zielbeschreibung	Wenn Objekte erstellt werden, gehören sie den ursprünglichen Autoren, bzw. den Teilnehmern des Treffens. Damit sie von anderen Teammitgliedern verwendet werden dürfen, müssen die Objekte mit dem Projekt / Team verknüpft werden.
Success End Condition	Das Objekt ist mit einem Team / Projekt verknüpft, und kann von Mitgliedern betrachtet und bearbeitet werden.
Failure End condition	Wie vor der Aktion
Primary Actor	Benutzer

Tabelle A.11.: Anwendungsfall: Objekte einem Projekt zuordnen

Anwendungsfall	**Metainformationen Editieren**
Zielbeschreibung	Objekte sollen mit weiteren Metainformationen angereichert werden, die nicht automatisch erfasst werden können, (Automatisch: Autoren und Zeitpunkte), oder automatisch erfasste Attribute sollen geändert werden. Dies nur sofern es Sinn macht.
Precondition	Vorhandenes Objekt
Success End Condition	Vorhandenes Objekt ist mit Metadaten angereichert.
Failure End condition	Wie vor der Aktion
Primary Actor	Benutzer

Tabelle A.12.: Anwendungsfall: Metainformationen editieren

Glossar

System Hier: Die entwickelte Lösung, zur Erweiterung der Möglichkeiten in einem Konferenzraum.

BEACH BEACH ist ein objektorientiertes Framework, daß die Infrasturktur für Roomwaresysteme zur Verfügung stellt.

Blackboard Eine Architektur einer Kommunikationsplattform für Prozesse. Alle teilnehmenden Prozesse können gleichberechtigt Nachrichten auf das Blackboard schreiben und von dort lesen. Siehe Erman u. a. (1980), oder Abschnitt 4.4.2.1.

COAST Ein in Smalltalk implementiertes Framework für verteilte Objekte. (Schümmer u. a. 2000)

Concurrent Versions System Ein Versionskontrollsystem für Verzeichnisse und Dateien.

CSCW CSCW bezeichnet [...] die theoretischen Grundlagen bzw. die Methodologien für Gruppenarbeit und deren Unterstützung durch Rechner.[1] (Schlichter u. a. 2001, 1)

CVS Siehe Concurrent Versions System

Dissappearing Computer Die Integration von Informationstechnik in Alltagsgegenständen. (Siehe auch: Ubiquitous Computing und Pervasive Computing)

Filiale Eine anwendungsneutrale Bibliothek, die die Funktionen des Betriebssystems und der Middleware kapselt und einer Anwendung zur Verfügung stellt.

HCI Siehe Human-Computer-Interaction

Human–Computer–Interaction Der englische Begriff für das Feld der Mensch-Computer-Interaktion

iRoom Ein mit UbiComp-Technologien aufgebauter interaktiver Gruppenarbeitsraum. Ein Teil des Interactive Workspaces-Projekts der Universität Stanford.

[1] `http://wwwfgcscw.in.tum.de/index.html`

iRos Ein MetaOs, eine Abstraktionsschicht zwischen spezifischen Betriebssystemen und den Anwendungen des iRooms.

Large Interactive Public Display Ein großer Bildschirm in öffentlichen Räumen, der Möglichkeiten zur Interaktion bietet

LIPD Siehe Large Interactive Public Display

MCI Siehe Mensch-Computer-Interaktion

Mensch-Computer-Interaktion Zu den Themen der Mensch-Computer-Interaktion gehören alle Fragen der benutzergerechten Gestaltung von interaktiven Informatiksystemen in vielfältigen Anwendungsfeldern [2](Janneck)

Middleware Eine zusätzliche Abstraktionsschicht zwischen Anwendung und Netzwerkbetriebssystem, die eine höhere Abstraktion erlaubt.

Objekt ID Eine eindeutige Bezeichnung für ein Objekt, über die eine Filiale auf das Objekt zugreifen kann. Eine mögliche Form wären URIs

Pervasive Computing Pervasive Computing ist mit dem Begriff des Ubiquitous Computing verwandt. Es bezeichnet die Vision, daß die Umgebung mit Rechenleistung durchsetzt ist, die für bestimmte Dienste verantwortlich ist.

Roomware 1.: Roomware bezeichnet die Integration von Elementen eines Raumes mit Informations Technologie. (Streitz u. a. 1999)
2.: Eine konkrete Umsetzung von 1. durch das IPSI Ambiente Team des Fraunhofer Instituts Darmstadt.

Subversion Als Nachfolger von CVS entwickeltes Versionskontrollsystem.

SVN Siehe Subversion

Ubicomp Siehe Ubiquitous Computing

Ubiquitous Computing Die Vision des Ubiquitous Computing ist, daß Computer ein so selbstverständlicher Teil unsers Lebens werden, daß sie aus der Wahrnehmung verschwinden. Eingeführt wurde der Begriff von Mark Weiser in seinem Artikel im Scientific American. (Weiser 1991)

[2] `http://www.acm.org/sigchi/`
`http://www.mensch-computer-interaktion.de/`

Universal Resource Identifier A Uniform Resource Identifier (URI) is a compact string of characters for identifying an abstract or physical resource. (Berners-Lee u. a. 1998)

URI Siehe Universal Resource Identifier

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 28. Februar 2006	
Ort, Datum	Unterschrift