Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Alexandra Revout

Ein System für das kollaborative Bearbeiten von Dokumenten in mobilen Umgebungen

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Alexandra Revout

Ein System für das kollaborative Bearbeiten von Dokumenten in mobilen Umgebungen

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang Master Informatik
am Studiendepartment Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

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

Abgegeben am 19. März 2008

Alexandra Revout

Masterarbeit

Ein System für das kollaborative Bearbeiten von Dokumenten in mobilen Umgebungen

Stichworte

Kollaboratives Schreiben, asynchrones Schreiben, Gruppenarbeit, Computer Supported Cooperative Work, Peer-to-Peer-Systeme, Mobilität, dynamische Umgebungen

Kurzzusammenfassung

Kollaborative Erstellung von Dokumenten spielt eine bedeutende Rolle in der Gruppenarbeit. In den letzten Jahren hat das Thema der mobilen und spontanen kollaborativen Arbeit sehr an Bedeutung gewonnen. Diese Entwicklungsrichtung des kollaborativen Schreibens fordert neue Lösungen für kollaborative Editiersysteme in Bezug auf Systemarchitektur und -Verhalten.

Im Rahmen dieser Arbeit wurde ein System-Konzept entwickelt, das eine leichtgewichtige Unterstützung der spontanen kollaborativen Arbeit an Dokumenten bietet. Das System ermöglicht eine asynchrone gemeinsame Bearbeitung von Dokumenten in einer Arbeitsgruppe. Basierend auf einer dezentralen Lösung und einer Peer-to-Peer-Kommunikation unterstützt das System eine dynamische Gruppenbildung und ermöglicht das Bearbeiten von gemeinsamen Dokumenten zur jeder Zeit und an jedem Ort.

Alexandra Revout

Title of the paper

A System for collaborative Document Editing in mobile Environments

Keywords

Collaborative writing, asynchronous editing, group work, Computer Supported Cooperative Work, peer-to-peer-systems, mobility, dynamic environments

Abstract

Collaborative preparing of documents is an important part in teamwork. Recently the topic of mobile and spontaneous collaborative work has gained more and more importance. The direction of this development of collaborative writing requires new solutions concerning the architecture and the properties of collaborative editing systems.

In this work a systematic concept has been developed presenting a lightweight support for spontaneous collaborative work with documents. This system allows an asynchronous common editing of documents in a working group. Based on a decentralized solution and a peer-to-peer-communication said system supports a dynamic grouping and enables the editing of common documents anytime and anywhere.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Die in den letzten Jahrzehnten passierten sozialen Veränderungen im Arbeitsumfeld – wie z.B. die zunehmende Globalisierung der wirtschaftlichen Märkte, die Entwicklung von neuen Organisationsformen und die Steigerung der Komplexität und der Dynamik der Aufgaben – förderten die zunehmende Verbreitung der kollaborativen Gruppenarbeit. Auf der anderen Seite erhöhten die großen technischen Fortschritte in Technologien für Computer-Hardware und -Software und Netzwerk-Infrastrukturen die Nachfrage nach einer stärkeren Computer-Integration in die Gruppenarbeit (nach [Gross]). Diese Trends waren einer der Gründe der Entstehung des interdisziplinären Forschungsbereichs von Computer Supported Cooperative Work (CSCW) (nach [Gross]). CSCW beschäftigt sich mit den theoretischen Grundlagen und Methodologien für Gruppenarbeit und ihre Computerunterstützung [Borghoff].

Kollaborative Erstellung von Dokumenten spielt eine bedeutende Rolle in der Gruppenarbeit: Dokumente sind meist verbreitete Artefakte, die bei der kollaborativen Arbeit benutzt und ausgetauscht werden [Yushun]. Das Verfassen eines Dokumentes von mehreren Personen verkürzt die Erstellungszeit durch die Aufteilung der Aufgaben und verbessert die Qualität des Dokumentes durch das gegenseitige Ergänzen des Inhaltes. Auf der anderen Seite kann es bei dem gemeinsamen Schreiben an einem Dokument zu Stil- und Informationsbrüchen kommen. Außerdem beeinflussen sich die Aktionen der einzelnen Autoren gegenseitig und können einander stören. Computersysteme, die für die kollaborative Dokumenten-Erstellung konzipiert werden, versuchen diese Probleme zu lösen und den Benutzern das gemeinsame Verfassen von

Dokumenten zu erleichtern. Diese Systeme gehören zum Bereich des Workgroup Computing, einer der System-Klassen von CSCW. Beim Workgroup Computing steht vor allem die Unterstützung der engen Kooperation der Gruppenmitglieder im Vordergrund.

Wichtigste Objekte bei der kollaborativen Dokumentbearbeitung sind Gruppen und Dokumente, mit denen Gruppen agieren. Gruppenmitglieder können geografisch schwach oder stark verteilt sein. Sie können an den Dokumenten synchron oder auch zeitversetzt arbeiten. Die Ausprägung der kollaborativen Arbeit hat eine direkte Auswirkung auf die Eigenschaften des Systems, das diese Arbeit unterstützen soll.

Sehr verbreitet ist ein zentraler Design-Ansatz für kollaborative Editiersysteme. Bei diesem Ansatz existiert eine zentrale Stelle, die die Verwaltung von Gruppen, Gruppenmitgliedern und Dokumenten übernimmt. Als ein Beispiel für eine zentralisierte Lösung kann die Metapher vom *Collaborative Workplace* genannt werden, die für eine Raum-bezogene kollaborative Arbeit, unter anderem Konferenzen, Meetings und auch das Erstellen von gemeinsamen Dokumenten, mit dem Einbezug der kollaborativen Werkzeuge wie z.B. Smartboard oder Blueboard [Mund] steht.

In den letzten Jahren hat das Thema der mobilen kollaborativen Arbeit sehr an Bedeutung gewonnen, nicht zuletzt durch steigende technische Möglichkeiten von mobilen Geräten wie Notebook und PDA und durch Fortschritte der drahtlosen Kommunikationstechnik. Der Aspekt spontaner Kollaboration spielt in letzter Zeit auch eine bedeutende Rolle: Die Zusammenarbeit kann überall stattfinden, wo dazu gerade die Mittel vorhanden sind. Dieser Trend ist auf die starke Entwicklung des Pervasive Computing zurückzuführen [Mattern]. Diese Entwicklungsrichtung des kollaborativen Schreibens fordert neue Lösungen für kollaborative Editiersysteme in Bezug auf Systemarchitektur und -Verhalten.

1.2 Zielsetzung

Im Rahmen dieser Masterarbeit wird ein Konzept für ein kollaboratives Editiersystem entwickelt, das eine leichtgewichtige Unterstützung der spontanen Kollaboration in dynamischen Umgebungen bietet. Das System wird eine asynchrone Bearbeitung von gemeinsamen strukturierten Dokumenten in einer Gruppe ermöglichen, unabhängig von einem Ort und dezentral. Dafür wird ein Peer-to-Peer-Ansatz für die Systemarchitektur verwendet. Das System soll folgende Funktionalität aufweisen:

- Verwaltung von kollaborativen Gruppen:
  Das System soll ermöglichen, eine kollaborative Gruppe zu erstellen, ihr beizutreten und sie zu verlassen.
- Bearbeitung von Dokumenten:
  Den Benutzern soll die Möglichkeit zur Verfügung stehen, neue Dokumente zu erstellen und Bearbeitungsaufgaben innerhalb einer Gruppe zu verteilen. Das System soll außerdem die Synchronisierung der unterschiedlichen Versionen eines Dokumentes gewährleisten.
- Zugriffsrechte-Verwaltung:
  Das System soll die Vergabe von Editierrechten und die Kontrolle der Zugriffe auf die gemeinsamen Dokumente unterstützen.

Des Weiteren soll ein strukturiertes Format für die gemeinsamen Dokumente ausgearbeitet werden, das eine fein-granulare Verteilung der Aufgaben ermöglichen wird.

Für die Peer-to-Peer-Kommunikation soll ein existierendes Framework eingesetzt werden.

Für das eigentliche Editieren der Dokumente soll auf ein existierendes und bewährtes Textberarbeitungsprogramm zurückgegriffen werden.

Darüber hinaus wird ein Prototyp des Editiersystems realisiert, der zum Nachweis der Tragfähigkeit des entworfenen Konzeptes dienen soll.

1.3 Gliederung der Arbeit

Im Kapitel „Grundlagen“(2) werden Konzepte und Vorteile von strukturierten Dokumenten und bekannte offene strukturierte Formate vorgestellt. Des Weiteren wird eine Einführung in interdisziplinäre Informatik-Bereiche des Dokumentenmanagements und CSCW gegeben. Darüber hinaus werden wichtigste Eigenschaften und Konzepte der kollaborativen Editiersysteme erläutert. Anhand einigen Forschungsprojekten werden Trends im Bereich des mobilen und pervasiven kollaborativen Schreibens vorgestellt.

Im Kapitel „Analyse“(3) wird die Vision dieser Masterarbeit vorgestellt. Darüber hinaus werden anhand eines Beispielszenarios die wichtigsten Anwendungsfälle und Eigenschaften des zu entwickelnden Systems analysiert.

Das Kapitel „Design und Realisierung“(4) präsentiert den Entwurf des Systems. Es werden die ausgearbeitete Dokumentenstruktur und Editierrechte-Strategie vorgestellt. Des Weiteren werden die Architektur des Systems und

der Design ihrer einzelnen Module beschrieben. Im Abschnitt „Realisierung“ werden interessante Aspekte der Prototyp-Realisierung vorgestellt und anhand eines Beispielablaufs die Funktionsweise des Systems beschrieben. Im Abschnitt „Evaluation“ wird das entwickelte Konzept bewertet.

Das Kapitel „Zusammenfassung und Ausblick“(5) enthält das Fazit über die durchgeführte Arbeit und über die Richtung der aktuellen Entwicklungen im Bereich der Kooperationsunterstützung der Zusammenarbeit.

Kapitel 2

Grundlagen

In diesem Kapitel wird eine kurze Einführung in die Themen des Dokumentenmanagements und des CSCW (Computer Supported Cooperative Work) und die Einordnung der kollaborativen Bearbeitung von Dokumenten in diese zwei Bereiche gegeben. Es werden die wichtigsten Konzepte des gemeinsamen Verfassens von Dokumenten erläutert und relevante Design-Ansätze und Technologien für die Dokumentenstruktur, Replikation der Dokumente und Architekturen vorgestellt. Im Weiteren wird ein Einblick in die aktuellen Forschungs- und Entwicklungstrends in diesem Bereich präsentiert.

2.1 Strukturierte Dokumente

Beim kollaborativen Bearbeiten von Dokumenten stehen vor allem die Dokumente selbst im Mittelpunkt. Aus diesem Grund spielt beim Design eines kollaborativen Editiersystems die Struktur der zu editierenden Dokumente eine bedeutende Rolle. Ein lineares Dokument, wie z.B. ein MS Word Dokument, besteht üblicherweise aus einem einzigen Teil bzw. einer Datei. Obwohl es inhaltlich eine Baumstruktur haben kann (Kapitel, Unterkapitel usw.), ist ein solches Dokument entweder mühsam oder überhaupt nicht in mehrere physikalische Teile trennbar. Derartige Dokumentenstruktur wirkt sich negativ auf den Kollaborationsprozess aus, da das ganze Dokument entweder für die Bearbeitung gesperrt oder jedes Mal auf die Inkonsistenzen bzw. Abweichungen der verschiedenen Versionen geprüft werden muss. Das kann dazu führen, dass das gemeinsame Dokument nur sequentiell von mehreren Personen bearbeitet werden kann oder die Ausführungszeiten der einzelnen

Operationen bei großen Dokumenten erheblich verlängert werden. Außerdem besteht die Gefahr einer hohen Anzahl von Bearbeitungskonflikten.

Eine hierarchische Dokumentenstruktur ist dagegen besser für das kollaborative Arbeiten geeignet, da das Dokument mit solcher Struktur auf mehrere physikalische Bereiche unterteilt werden kann, die unabhängig voneinander behandelt werden können. Die einzelnen Teile des Dokumentes können dabei ohne Konflikte parallel bearbeitet werden.

Die Verwendung einer logischen und hierarchischen Struktur für Dokumente ist schon seit den 80er Jahren des letzten Jahrhunderts ein etabliertes Thema ([Borghoff]). Zu den Vorreitern von strukturierten Formaten kann vor allem die Standard Generalized Markup Language (SGML) gezählt werden, die 1986 als ein ISO-Standard herausgebracht wurde. Das Konzept von SGML schrieb in erster Linie die Trennung zwischen der formalen Beschreibung des Dokument-Layouts und seinem Inhalt vor [Mintert]. Darüber hinaus ermöglichte SGML einen hohen Strukturierungsgrad der Dokumente und die Trennung zwischen der logischen Dokumentenstruktur und der physikalischen Speicherung [Borghoff]. Obwohl SGML den ganz großen Durchbruch nicht schaffte, fanden ihre Untermengen wie die Metasprache XML große Akzeptanz und sind weit verbreitet.

Zu den bekanntesten Anwendungen von SGML, die auch aktuell weiter entwickelt werden, zählt neben HTML das Dokumentenformat DocBook [DocBook]. DocBook wurde ursprünglich zur Erstellung von Hardware- und Software-Dokumentationen konzipiert. Wie auch SGML und XML gehört DocBook zu den *Descriptive Markup Languages* (DML)[1]. Das DocBook-Format wird durch eine DTD (Document Type Definition) beschrieben, obwohl in der Planung auch die Entwicklung eines XML-Schemas steht. Ursprünglich war es ein reines SGML-Format, aber seit 1997 wird auch ein XML-DocBook entwickelt, so dass aktuell DocBook sowohl in einer SGML- als auch in einer XML-Version vorhanden ist. Aktuell ist für die Entwicklung und die Pflege von DocBook das DocBook Technical Committee bei OASIS [OASIS DocBook TC] verantwortlich. DocBook wird besonders für die Erstellung von technischen Büchern und Dokumentationen eingesetzt.

Des Weiteren ist die Textsatz-Anwendung LaTeX ([Lamport]) sehr verbreitet. LaTeX ist ein Aufsatz für TeX, einem sprachlich umfangreichen Satzsystem, und wurde Anfang der 1980er von Leslie Lamport zur Vereinfachung der Arbeit mit TeX entwickelt. LaTeX, wie auch TeX, basiert auch auf den

[1] DML sind Auszeichnungssprachen, die die Informationen und Daten beschreiben (nach [IT Wissen]). Sie fügen dem Text die Informationen über die Struktur des Dokumentes zu, sagen aber nichts über die Formatierung aus (nach [Holzinger]).

Grundkonzepten von SGML, obwohl sie zu den *Procedural Markup Languages* (PML)[2] gehört und ihr ein anderer Ansatz zugrunde liegt: In LaTeX wird Text mit Hilfe von vordefinierten Befehlen (Makros) ausgezeichnet. Dieses Format findet große Verbreitung beim Verfassen von verschiedenen Arten von Schriftstücken wie Artikeln, Büchern etc. Vor allem wird LaTeX im Bereich der Universitäten und Hochschulen zum Erstellen von wissenschaftlichen Arbeiten und beim Herausgeben von Artikelsammlungen und Kompendien, die durch mehrere Autoren verfasst werden, eingesetzt.

Die zwei oben beschriebenen Formate dienen zur Verfassung von Texten, in der Regel durch eine Person. Ein kollaboratives Dokument stellt aber eine viel komplexere Struktur dar, die durch mehrere Personen bearbeitet wird und abgesehen von dem eigentlichen Text auch hierarchische Meta-Informationen enthalten soll. Dazu gehören z.B. die Informationen über den Autor, die Editierrechte, Datum der Erstellung und der letzten Änderung usw. Für die Beschreibung dieser Struktur ist die Metasprache XML [XML] besonders gut geeignet. XML ist ein plattformunabhängiger Standard und wurde durch das World Wide Web Consortium (W3C) spezifiziert. Auf der Basis von XML können weitere anwendungsspezifische Sprachen definiert werden. Diese Definitionen können entweder durch eine DTD oder durch ein XML-Schema beschrieben werden. Einer der größten Vorteile von XML liegt in der hierarchischen Strukturierung. Dementsprechend können einzelne Teile des Dokumentes leicht als Teilbäume zwischen den Gruppenmitgliedern übertragen und eingebunden werden. Dabei stellt die Entwicklung eines effizienten Verfahrens zur Synchronisierung von XML-Bäumen eine besondere Herausforderung beim Design eines kollaborativen Editiersystems dar. In weiteren Abschnitten werden einige Forschungsprojekte vorgestellt, in denen eventuelle Lösungen für dieses Problem vorgeschlagen wurden (siehe [Neyem, Mascolo]). Weitere innovative Ideen können unter anderem in [Lindholm] und [Fontaine] nachgelesen werden.

Wie auch DocBook und LaTeX ist XML ein Text-Format und damit auch für Menschen prinzipiell lesbar. Dementsprechend ist kein spezieller Editor zur Bearbeitung von Dokumenten in solchen Formaten unbedingt nötig und im schlimmsten Fall kann dafür auch ein einfacher Text-Editor eingesetzt werden. Der weitere große Vorteil eines Text-Formates liegt darin, dass die Daten, in diesem Fall Dokumente, für die Übertragung stark komprimiert werden können. Das gestattet eine deutliche Verringerung des zu übertra-

[2]PML sind Auszeichnungssprachen, die die Verfahren für die Darstellung und Formatierung des Dokumentes beschreiben (nach [IT Wissen]). Bei einigen PML sind auch die Informationen über die Struktur des Textes enthalten (nach [Holzinger]).

genden Datenvolumens, was vor allem in mobilen Umgebungen von Vorteil ist.

2.2 Dokumentenmanagement und CSCW

Das kollaborative Bearbeiten von Dokumenten gehört zu den Teilaspekten solcher großen interdisziplinären Informatik-Bereiche wie Dokumentenmanagement und Computer Supported Cooperative Work (CSCW). In diesem Abschnitt werden diese zwei Forschungsgebiete kurz vorgestellt. Darüber hinaus wird das Thema der kollaborativen Dokumentenbearbeitung in diese Bereiche eingeordnet.

2.2.1 Kollaboratives Schreiben im Dokumentenmanagement

Der Begriff Dokumentenmanagement umfasst verschiedene Aspekte der digitalen Verwaltung von Dokumenten. Dabei werden zwei Bedeutungen dieses Begriffes unterschieden: Dokumentenmanagement im engeren Sinn und Dokumentenmanagement im weiteren Sinn [Kampffmeyer]. Unter dem Dokumentenmanagement im engeren Sinn, auch klassisches Dokumentenmanagement genannt, werden im Wesentlichen Compound Document Management[3] und dynamische Ablagesysteme zur Verwaltung des Lebenszyklus von Dokumenten verstanden. Diese Systeme haben solche Eigenschaften wie visualisierte Ordnungsstrukturen, Versionierung und datenbankgestützte Metadatenverwaltung zur Indizierung und Suche von Dokumenten.

Unter dem Dokumentenmanagement im weiteren Sinn werden dagegen mehrere verschiedene Systemkategorien und ihr Zusammenspiel verstanden. Dazu gehören unter anderem klassisches Dokumentenmanagement, Bürokommunikation, Document Imaging (Digitalisierung von Papierdokumenten), Workflow, Groupware und elektronische Archivierung. Bei diesen Kategorien steht jeweils ein anderer Aspekt der Dokumentenverwaltung im Mittelpunkt. In der Abbildung 2.1 sind verschiedene Betrachtungswinkel in Bezug auf das Dokumentenmanagement dargestellt. Unter dem Blickwinkel „Dokument“ wird das Dokumentenmanagement im klassischen Sinn verstanden, d.h. Zugriff, Kontrolle und Verwaltung erfolgen auf der Basis von Dokumentenmerkmalen.

[3] Compound Document Management - Verwaltung von „aus beliebigen Objekten wie Text, Bild, Tabelle, Audio, Video etc. zusammengesetzten Dokumenten“ (vgl. [Kampffmeyer]).

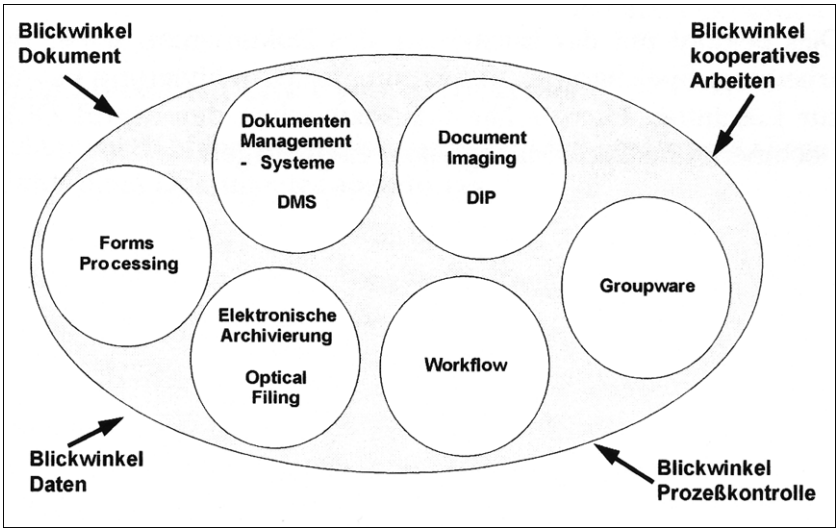

Abbildung 2.1: Aspekte des Dokumentenmanagements [Kampffmeyer]

Beim Schwerpunkt „Prozesskontrolle“ steht dagegen die Einbindung von Dokumenten in Arbeitsabläufe im Mittelpunkt. Dazu gehören Workflow- und Businessprozess-Managementsysteme. Der Ansatz „Daten“ betrachtet wiederum nicht die einzelnen Dokumente, sondern die darin enthaltenen Daten. Von diesem Ansatz gehen besonders volltextorientierte und Archivierungssysteme aus.

Bei dem „Kooperativen Arbeiten“ stehen die Kommunikation und die gemeinsame Nutzung von Informationsressourcen innerhalb einer Gruppe im Vordergrund. Zu den Systemen mit diesem Schwerpunkt gehören in erster Linie Groupware-Systeme (siehe Abbildung 2.1). Als Groupware werden Systeme bezeichnet, die die Arbeit von mehreren Personen innerhalb einer Gruppe unterstützen. Unter anderem gehört zu den Aufgaben von Groupware-Systemen auch die Unterstützung der kollaborativen Bearbeitung von gemeinsamen Dokumenten. Die Entwicklung solcher Systeme integriert die theoretischen Grundlagen, die im Rahmen von CSCW spezifiziert wurden (nach [Borghoff]). Im nächsten Abschnitt wird auf das Thema CSCW ausführlicher eingegangen.

2.2.2 Einordnung des kollaborativen Schreibens ins CSCW

Computer Supported Cooperative Work ist ein umfangreicher interdisziplinärer Forschungsbereich, der sich mit den theoretischen Grundlagen und Methodologien für Gruppenarbeit und ihre Computerunterstützung beschäftigt [Borghoff]. Für CSCW-Systeme existieren drei wesentliche Klassifizierungs-Schemata: nach der Raum-Zeit-Matrix, nach funktionellen Aspekten und nach dem so genannten 3K-Modell.

Die Klassifizierung nach Raum und Zeit unterscheidet räumliche und zeitliche Verteilung der beteiligten Gruppenmitglieder. Die Personen können sich entweder im selben Raum oder im gleichen Gebäude, in verschiedenen Städten usw. befinden. Außerdem können sie miteinander in Realzeit (synchron) oder zeitversetzt (asynchron) kommunizieren. Je nach Raum- und Zeit-Verteilung unterscheiden sich folglich die Eigenschaften der Zusammenarbeit. In der Abbildung 2.2 sind beispielhaft einige Arten der Zusammenarbeit nach zweidimensionaler Unterscheidung nach Raum und Zeit aufgeteilt worden. Systeme, die diese Arbeit und Interaktionen zwischen den Gruppenmitgliedern unterstützen, müssen dementsprechend unterschiedliche Merkmale aufweisen und können so klassifiziert werden.

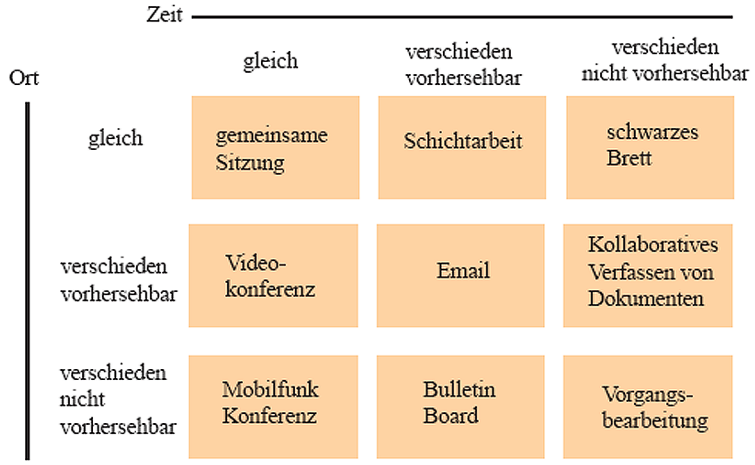

Abbildung 2.2: Raum-Zeit-Matrix [Schlichter]

Bei der Klassifizierung nach Funktionen steht die Anwendungsorientierung im Vordergrund. Nach [Ellisb] können folgende funktionelle Kategorien von CSCW-Systemen unterschieden werden:

- **Nachrichtensysteme:** Diese Systeme unterstützen den Austausch von textuellen Nachrichten zwischen den Gruppenmitgliedern. Dazu gehören vor allem E-Mail und Foren-Systeme, aber auch Blogs können dazu gezählt werden.
- **Gruppeneditoren:** Sie ermöglichen das gemeinsame Verfassen von Dokumenten in einer Gruppe. Nach der Arbeitsweise können Gruppeneditoren in Realzeit-Editoren und asynchrone Editoren unterteilt werden.
- **Gruppen-Entscheidungsunterstützungssysteme:** Diese Systeme bieten Hilfsmittel zur Erforschung von unstrukturierten Problemen in einer Gruppe und unterstützen eine iterative Entscheidungsfindung. Viele solche Systeme werden als elektronische Sitzungsräume umgesetzt, die mehrere vernetzte Rechner, große öffentliche Displays und Audio- und Video-Ausstattung beinhalten.
- **Konferenzsysteme:** Sie werden nach der Art der angebotenen Leistungen in drei Kategorien unterteilt: Systeme, die Echtzeit-Konferenzen, Telekonferenzen und Desktop-Konferenzen unterstützen.
- **Agentensysteme:** Intelligente Agenten werden als aktiver Bestandteil der Gruppenarbeit z.B. in Computerspielen und zur Überwachung von Sitzungen im Sinne eines Moderators eingesetzt.
- **Koordinationssysteme:** Diese Systeme dienen dazu, die Ergebnisse der Arbeit einzelner Personen zur Vollendung eines gemeinsamen Zieles zu integrieren und zu adaptieren. Das Problem der Koordination tritt überwiegend bei einer asynchronen Gruppenarbeit auf.

Bei dem dritten Klassifizierungsmodell werden CSCW-Systeme nach ihrer Haupt-Unterstützungsfunktion charakterisiert. Je nach der Art der Zusammenarbeit in einer Gruppe werden drei wesentliche Unterstützungsfunktionen unterschieden. Diese bilden das 3K-Modell, das nach den Anfangsbuchstaben der entsprechenden Begriffe benannt wurde (nach [Borghoff]):

- Kommunikationsunterstützung: Unterstützung der Verständigung von Personen mittels Informationsaustausch.
- Koordinationsunterstützung: Unterstützung der Abstimmung aufgabenbezogener Aktivitäten und Ressourcen.
- Kooperationsunterstützung: Unterstützung der Verfolgung von gemeinsamen Zielen.

Dieses Modell wird typischerweise als ein Dreieck dargestellt (siehe Abbildung 2.3). Je nach dem Grad der Unterstützung der einer oder der anderen Funktion können die funktionellen Kategorien von CSCW-Systemen in diesem Dreieck entsprechend platziert werden (nach [Teufel]). Im Zusammen-

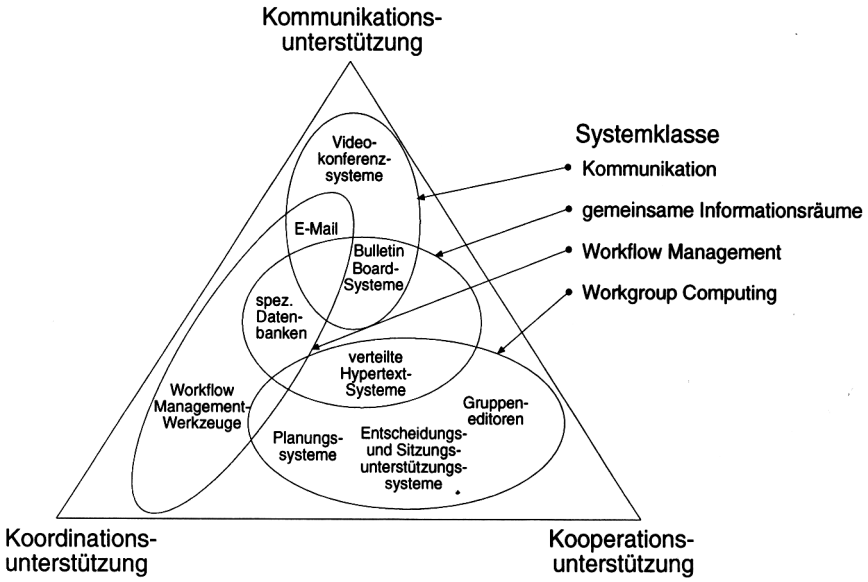

Abbildung 2.3: Klassifikationsschema nach Unterstützungsfunktionen [Teufel]

hang mit dem Grad der Ausprägung der einen oder anderen Unterstützungsfunktion werden CSCW-Systeme nach vier Klassen unterteilt: Kommunikationssysteme, Gemeinsame Informationsräume, Workflow Management und Workgroup Computing (nach [Schlichter]). Kommunikationssysteme ermöglichen den expliziten Informationsaustausch zwischen den Gruppenmitgliedern. Dazu gehören in erster Linie Nachrichten- und Konferenzsysteme, z.B. Instant Messaging-Systeme ([ICQ, Windows Life]). Die modernen Instant Messenger bieten abgesehen von Echtzeit-Kommunikation der Teilnehmer auch Video- und Telefonkonferenzen und Datei-Austausch. Obwohl solche Messenger auch als eine Art des kollaborativen Schreibens angesehen werden können, unterstützen sie keine kollaborative Bearbeitung von Dokumenten.

Zur Klasse Gemeinsame Informationsräume gehören Systeme, die einen impliziten Informationsaustausch in der Gruppe ermöglichen. Zu den Aufgaben von solchen Systemen zählen Verwaltung und Speicherung von gemeinsamen Informationen und Bereitstellung von geeigneten Zugriffsmechanismen auf diese Informationen. In den letzten Jahren ist die Entwicklung in diesem

Bereich stark in die Richtung von Browser-basierten Informationssystemen gegangen. Als Beispielsystem kann hier [BSCW] genannt werden. Dieses System unterstützt vor allem die asynchrone Bearbeitung und die Versionierung der gemeinsamen Dokumente. Darüber hinaus bietet es dem Benutzer solche Funktionalität wie das Einladen der anderen Benutzer zur Mitarbeit an einem Dokument, Rechtevergabe usw.

Beim Workflow Management steht die Koordination der Aktivitäten der einzelnen Gruppenmitglieder und ihrer Ressourcen im Mittelpunkt.

Das Workgroup Computing hat als Ziel die flexible Unterstützung der engen Kooperation zwischen den Gruppenmitgliedern, insbesondere bei schwach strukturierten und sich selten wiederholenden Aufgaben. Zu solchen Aufgaben gehören unter anderem die Entscheidungsfindung in der Gruppe wie auch kooperative Erstellung und Bearbeitung von Dokumenten, die auch unter dem Begriff *Kollaboratives Schreiben* (auch Kooperatives Schreiben) verstanden wird. Als Beispiele für Systeme dieser Klasse können Gruppenentscheidungssysteme, die Face-to-Face-Sitzungen unterstützen, und Gruppeneditoren genannt werden [Borghoff].

Diese Masterarbeit beschäftigt sich mit dem kollaborativen Bearbeiten von gemeinsamen Dokumenten. Entsprechend der beschriebenen Klassifizierung nach dem 3K-Modell gehört das in dieser Arbeit zu behandelnde Thema zum Bereich von Workgroup Computing.

Zu einer besonderen Ausprägung der kollaborativen Erstellung von Texten können Wikis gezählt werden, die in der letzten Zeit eine große Popularität erlangt haben. Nach [Ebersbach] kann der Begriff Wiki folgendermaßen definiert werden:

> „Ein Wiki ist eine webbasierte Software, die es allen Betrachtern einer Seite erlaubt, den Inhalt zu ändern, indem sie diese Seite online im Browser editieren. Damit ist das Wiki eine einfache und leicht zu bedienende Plattform für kooperatives Arbeiten an Texten und Hypertexten."

Wikis erlauben demnach ein asynchrones kollaboratives Bearbeiten von Online-Texten. Eventuelle Bearbeitungskonflikte werden dabei entweder durch das Sperren der gerade bearbeiteten Seiten ([TWiki]) oder durch einen in das Wiki-System integrierten Überwachungs-Mechanismus ([MediaWiki]), der die Konflikte entdeckt und darauf hinweist, vermieden [Ebersbach]. Das wichtigste Konzept von Wikis liegt in der Versionsverwaltung. Wie auch Web-basierte Gemeinsame Informationsräume bauen Wikis auf zentralisierter Client-Server-Architektur auf. Der Einsatzgebiet von Wikis hat sich in

den letzten Jahren sehr erweitert. Sie können als ein webbasiertes Content Management System zur Bearbeitung und Verwaltung von Internetauftritten genutzt oder für Diskussionsforen oder Brainstorming verwendet werden. Darüber hinaus werden auch einige Versuche gestartet, Wikis als Projektmanagement-Werkzeug für die Begleitung von unterschiedlichen Projektphasen einzusetzen, angefangen von Projektkonzeption, Projektplanung und Präsentation der Ergebnisse bis zur Koordinierung der Projektteilnehmer ([Ebersbach]). In erster Linie aber werden Wikis als Wissensmanagement-Werkzeug verwendet, für das Sammeln und Verwalten sowohl des allgemeinen, enzyklopädischen (z.B. die Online-Enzyklopädie Wikipedia) als auch des fachlichen Wissens, z.B. der firmeninternen Dokumentationen und Einleitungen. Obwohl Wikis das gemeinsame, asynchrone Schreiben von Texten ermöglichen, können sie nicht zu den Systemen für die kollaborative Erstellung von Dokumenten zählen, da sie nicht dazu gedacht sind, Dokumente im klassischen Sinn zu produzieren. Sie können eher als ein öffentlich oder eingeschränkt zugänglicher Speicher von Informationen betrachtet werden.

2.3 Wichtigste Konzepte des kollaborativen Schreibens

In diesem Abschnitt wird eine allgemeine Klassifikation der Systeme für kollaboratives Bearbeiten von Dokumenten präsentiert. Darüber hinaus werden wichtigste Eigenschaften der kollaborativen Editiersysteme erläutert und Konzepte der relevanten Systemarchitekturen vorgestellt.

2.3.1 Klassifizierung der kollaborativen Editiersysteme

Das kollaborative Schreiben kann in synchrones, asynchrones und hybrides Schreiben unterteilt werden. Beim synchronen Schreiben, auch unter dem Begriff Echtzeit-Editieren bekannt, ist vor allem die sofortige Sichtbarkeit von Aktionen der einzelnen Gruppenmitglieder von großer Bedeutung. Systeme, die solches Editieren unterstützen, arbeiten nach dem WYSIWIS-Prinzip (What You See Is What I See). Hier sind keine Verzögerungen in der Aktualisierung akzeptabel. Bei dieser Art von Kollaboration ist die Nebenläufigkeitskontrolle von besonderen Bedeutung [Borghoff].

Bei dem asynchronen kollaborativen Schreiben wird keine Parallelität der Aktionen angestrebt. Die individuellen Arbeitsschritte der einzelnen Grup-

penmitglieder passieren in der Regel zu verschiedenen Zeiten. Verzögerungen bei der Aktualisierung des Dokumentzustandes sind daher eher unwichtig. Die von einem Teilnehmer durchgeführten Text-Modifikationen werden nicht gleich an andere propagiert, sondern zu einem beliebigen Zeitpunkt als eine neue Version des Dokumentes. Das geschieht entweder durch eine explizite, von dem Teilnehmer initiierte, Aktion oder „bei Gelegenheit“, wenn z.B. das Gruppenmitglied wieder online ist. Der Schwerpunkt des asynchronen Editierens liegt in der Unterstützung und der Verwaltung von Zugriffen auf die gemeinsam genutzten Dokumente, in der Synchronisation und Zusammenführung (Merging) von verschiedenen Versionen eines Dokumentes und in der Erkennung und Behebung der Konflikte [Borghoff].

Beim hybriden Konzept handelt es sich um Systeme, die sowohl synchrones als auch asynchrones kollaboratives Schreiben unterstützen, in Abhängigkeit von aktuellen Bedingungen, z.B. ob der Teilnehmer online ist oder nicht.

2.3.2 Eigenschaften der kollaborativen Editiersysteme

Die kollaborativen Editiersysteme zeichnen folgende wichtigste Eigenschaften aus (nach [Schwabe, Borghoff]):

- *Kooperationsunterstützung*: Funktionalitäten, die gemeinsame Arbeitsobjekte (Dokumente) bereitstellen und gemeinsame und individuelle Sichten auf die Inhalte ermöglichen.
- *Kollaborative Awareness*: Die Wahrnehmung der anderen Gruppenmitglieder und ihrer Handlungen muss gegeben sein, was ein konsistentes Verständnis der gemeinsamen Aufgabe ermöglicht. Die Gruppenmitglieder sind dadurch nicht voneinander isoliert.
- *Nebenläufigkeitskontrolle*: Die Kontrolle von konkurrierenden Zugriffen auf die gemeinsamen Dokumente ist ein zentrales Thema im kollaborativen Schreiben. Es existieren mehrere Ansätze zur Lösung dieses Problems, die in weiteren Abschnitten erläutert werden.
- *Versionierung, Annotation*: Durch die Versionierung und Änderungsvermerke der einzelnen Gruppenmitglieder wird eine Historie der Bearbeitung eines Dokumentes aufgebaut, was einen wichtigen Teil zur Awareness beiträgt.
- *Zugriffskontrolle*: Durch die Vergabe von Zugriffsrechten wird innerhalb einer Arbeitsgruppe bestimmt, welche ihrer Mitglieder auf welche Dokumente einen lesenden bzw. schreibenden Zugriff haben sollen.

Bezüglich außerhalb der Gruppe wird garantiert, dass nur Mitglieder dieser Gruppe einen Zugriff auf die Dokumente haben können.

- *Benutzerverwaltung*: Das Hinzufügen oder Entfernen von Mitgliedern gewährleistet eine dynamische Gruppenarbeit.
- *Konflikterkennung*: Erkennung der Konflikte bei einem gleichzeitigen Zugriff von mehreren Personen auf ein Dokument bzw. Erkennung der Existenz von mehreren parallelen Versionen eines Dokumentes.

Beim synchronen Editieren sollen Änderungen an einem Dokument sofort an alle Beteiligten propagiert werden und sofort bei allen sichtbar sein. Bei diesem Ansatz ist keine Zusammenführung von verschiedenen Versionen eines Dokumentes nötig. Dafür kommen aber einige zusätzliche Anforderungen zu den schon beschriebenen dazu, die eingehalten werden sollen [Yang, Gerlicher]:

- *Hohe Ansprechbarkeit (Responsiveness)*: Die Antwortzeiten (die Zeit, bis eigene Benutzeraktionen lokal sichtbar werden) sollen keine erkennbaren Verzögerungen aufweisen. Die Benachrichtigungszeiten (die Zeit, in der Aktionen an die entfernten Benutzer propagiert werden) sollen kurz sein.
- *Bewahrung der Kausalität*: Voneinander abhängige Operationen sollen in ihrer natürlicher Kausalordnung durchgeführt werden.
- *Konvergenz der Ergebnisse*: Nach der Ausführung gleicher Menge von Operationen sollen keine Unterschiede zwischen den einzelnen lokalen Dokumentkopien existieren.
- *Intentionserhaltung*: Das eingetretene Ergebnis einer durchgeführten Operation entspricht dem erwarteten, d.h. bei der Ausführung von zwei parallelen Operationen an einem Objekt wird das Ergebnis jeder Operation nicht von der konkurrierenden Operation überschrieben.

Der synchrone Ansatz ist seit den 90er Jahren ein etabliertes Thema im Bereich der kollaborativen Anwendungen. Das bekannteste Verfahren für die Nebenläufigkeitskontrolle ist das Verfahren der Operationalen Transformationen, das auf solchen Algorithmen wie dOPT [Ellisa, Schlichter] oder TreeOPT [Ignat] basiert. Die erste Generation von synchronen kollaborativen Anwendungen zeichnet vor allem aus, dass sie für feste, drahtgebundene Netzwerk-Topologien entwickelt wurden. Bekannte Projekte in diesem Bereich sind unter anderem REDUCE [Yang] und GROVE [Ellisb].

In den letzten Jahren hat das Thema der mobilen kollaborativen Arbeit sehr populär geworden. Steigende technische Möglichkeiten von mobilen Geräten, wie Notebook und PDA, und Fortschritte der drahtlosen Kommunikationstechnik fördern das Bestreben von Menschen, kollaborativ, mit beliebigen Geräten und vor allem mobil zu arbeiten [Mattern]. Diese Entwicklungen fordern neue Lösungen für kollaborative Editiersysteme in Umgebungen mit besonderen Eigenschaften [Leone]. Dazu zählen in erster Linie Aspekte der mobilen Umgebungen, z.B. die Anfälligkeit des Signals für Störungen, niedrigere Übertragungs-Bandbreite und Wechsel der Access-Points, die eine bedeutende Auswirkung auf das Verhalten des Systems haben können. Zu der anderen Gruppe von Aspekten gehören spezifische Eigenschaften von mobilen Kleingeräten (PDAs). Solche Geräte zeichnen sich durch Einschränkungen in Bezug auf verfügbare Ressourcen und Darstellungsmöglichkeiten aus. Im Zusammenhang mit diesen Eigenschaften ist in letzter Zeit die Anpassungsfähigkeit des Verhaltens von kollaborativen Systemen an ihre Umgebung (*Adaptivity*) ein sehr aktuelles Thema geworden: Ein kollaboratives Editiersystem soll im Stande sein, sein Verhalten an sich ändernde Bedingungen dynamisch anzupassen, wie z.B. das Aufrechterhalten der kollaborativen Session bei Verbindungsabbruch und -wiederaufnahme [Yushun].

Starke Entwicklung des *Pervasive Computing* ist der Grund dafür, dass der Aspekt spontaner Kollaboration in letzter Zeit ebenfalls eine bedeutende Rolle spielt: Der Wunsch und die Nachfrage nach einer Zusammenarbeit überall, wo dazu gerade die Mittel vorhanden sind, steigt [Mattern]. Dies erfordert wiederum neue Lösungen in Bezug auf die Systemarchitektur und -verhalten.

2.3.3 Strategien bei der Nebenläufigkeitskontrolle

Die Kontrolle von konkurrierenden Zugriffen auf gemeinsame Objekte ist ein zentrales Problem der meisten CSCW-Systeme [Borghoff]. Es werden zwei Gruppen von Verfahren für die Nebenläufigkeitskontrolle unterschieden: optimistische und pessimistische [Borghoff].

2.3.3.1 Optimistische Verfahren

In CSCW-Systemen mit einer niedrigen Wahrscheinlichkeit von konkurrierenden Zugriffen und geringen Anforderungen an die Konsistenz der gemeinsamen Dokumente kann optimistische Nebenläufigkeitskontrolle eingesetzt werden [Borghoff]. Die Aufgabe der optimistischen Verfahren liegt nicht in der Vorbeugung von Inkonsistenzen, sondern in der Konfliktentdeckung und

der Konfliktauflösung. Die Benutzer können in solchen Systemen zu jedem Zeitpunkt auf das gemeinsame Dokument zugreifen und dessen Inhalt manipulieren. Im Gegensatz zu dem aus dem Datenbanken-Bereich bekannten Begriff Optimistic Concurrency Control [Kemper] garantieren optimistische Verfahren aus dem CSCW-Bereich keine Konsistenz der Daten. Bei eventuellen Konfliktsituationen werden die durchgeführten Modifikationen nicht zurückgesetzt, sondern als eine extra Version des Dokumentes gespeichert. Der Benutzer wird umgehend darüber informiert und kann anschließend seine Änderungen durch das Zusammenführen von zwei auseinander gelaufenen Versionen in das originale Dokument übertragen und eine neue aktuelle Version des Dokumentes erstellen. Da das üblicherweise mit dem zusätzlichen Aufwand verbunden ist, sind optimistische Verfahren eher für Systeme geeignet, die das asynchrone Editieren unterstützen, bei dem Gruppenmitglieder meistens unterschiedliche Teile des Dokumentes bearbeiten und deswegen nur wenige oder keine Konflikte entstehen können.

2.3.3.2 Pessimistische Verfahren

In CSCW-Systemen mit hohen Anforderungen an die Konsistenz der gemeinsamen Dokumente müssen konkurrierende Schreibzugriffe synchronisiert werden, wie bei verteilten Datenbanken. In diesen Systemen kommen deswegen pessimistische Verfahren zum Einsatz, die sich durch zentrale oder dezentrale Kontrolle auszeichnen. In den nächsten Abschnitten wird ein Überblick über die wichtigsten pessimistischen Verfahren für die CSCW-Systeme gegeben.

Zentrale Kontrolle

Verfahren mit ausgezeichneter Kontrolleinheit Bei diesen Verfahren existiert ein ausgezeichneter Knoten (Teilnehmer), der alle Zugriffe auf die Dokumente serialisiert und synchronisiert. Die Entdeckung und Auflösung von Konflikten ist bei diesen Verfahren einfach, die Herausforderung liegt dagegen in einem robusten und effizienten Konzept zur Erhaltung einer und nur einer ausgezeichneten Kontrolleinheit. Die Schwierigkeiten können dabei bei einem (kurzfristigen) Ausfall dieser ausgezeichneten Einheit auftreten. Ein Beispiel für ein solches Verfahren ist das sogenannte Primary-Site-Verfahren [Borghoff], bei dem ein Dokument zwar repliziert ist, aber nur ein bestimmter Knoten für dieses Dokument „hauptverantwortlich“ ist.

Token-Verfahren Bei diesen Verfahren ist keine ausgezeichnete Kontrolleinheit für ein Dokument verantwortlich, sondern ein entlang eines virtuellen, aus den beteiligten Teilnehmern bestehenden Ringes wandernder Token. Je-

der Teilnehmer erhält den Token nur innerhalb einer fest vorgegebenen Zeit. Der Teilnehmer, der den Token besitzt, hat die gleichen Rechte wie die ausgezeichnete Einheit im vorigen Verfahren: Er serialisiert und synchronisiert alle Zugriffe auf ein repliziertes Dokument, für das er zur Zeit zuständig ist. Anspruchsvolle Aufgaben sind hier der Aufbau des virtuellen Ringes angesichts der Bestimmung der nächsten Nachbarn und dynamische Rekonfigurierbarkeit des Ringes beim Zufügen und Entfernen (auch wegen eines Kommunikationsausfalls) von Teilnehmern.

Dezentrale Kontrolle

Beim dezentralen Ansatz gibt es keine zentrale Stelle, die die Koordination von allen Zugriffsoperationen durchführt. Die Kontrolle wird von allen Teilnehmern der Gruppe auf eine gleichberechtigte Weise übernommen. Im Folgenden werden einige ausgewählte Verfahren mit dezentraler Kontrolle ohne Votierung kurz vorgestellt (nach [Borghoff]):

Einfache Sperrverfahren Durch die Sperre bekommt ein Gruppenteilnehmer ein exklusives Recht auf die Bearbeitung einer Informationsressource. Dabei ist das Anfordern und das Setzen von Sperren mit einem zusätzlichen Zeitaufwand verbunden. Außerdem müssen zusätzliche Wartezeiten berücksichtigt werden für den Fall, dass die Sperre nicht sofort genehmigt werden kann. Darüber hinaus spielt die Sperrgranularität eine bedeutende Rolle für den erreichbaren Grad der Nebenläufigkeit: Werden Sperren auf ein ganzes Dokument oder auf seine Teile gesetzt? Allgemein ist dieses Verfahren nicht für das synchrone Editieren zu empfehlen, da bei einer zu groben Sperrgranularität kein paralleles synchrones Arbeiten möglich ist. Außerdem können durch die relativ große Zeit, die die Sperrung erfordert, Verzögerungen bei den einzelnen Benutzern entstehen, die den Arbeitsfluss stören [Gerlicher].

Floor-Passing sind Verfahren mit wechselnder Kontrolle. Zu einem Zeitpunkt besitzt nur ein Teilnehmer die Kontrolle über das Dokument und somit die Zugriffsberechtigung. Herausfordernd ist hier die Koordination der Kontrolle-Übergabe und faire Verteilung des Floor-Besitzes. Zu den Verfahren mit dezentraler Kontrolle, die diese Probleme lösen, zählt implizites Floor-Passing mit dezentraler Koordination. Allerdings existieren auch zentralisierte Ansätze, wie z.B. implizites Floor-Passing mit Koordinationsstelle, zu deren Aufgaben die Durchführung der Synchronisation gehört. Bei den impliziten Verfahren übernimmt das System die Kontrolle über das Anfordern und die Vergabe des *Floors*.

Transaktionsverfahren ist ein konventioneller Ansatz zur Konsistenzerhal-

tung der Daten und stützt sich auf Serialisierungs-Mechanismen, die aus dem Datenbank-Bereich bekannt sind (mehr dazu in [Kemper]). Diese Verfahren werden größtenteils bei der asynchronen kollaborativen Arbeit eingesetzt, da eine Transaktion unter Umständen relativ lange dauern kann, was bei synchronen Dokumentenbearbeitung zur Störung des Arbeitsablaufs führen kann [Borghoff]. In CSCW-Systemen finden sowohl pessimistische als auch optimistische Transaktionsverfahren eine Verwendung.

Operationale Transformationen sind ein etablierter pessimistischer Ansatz zur Nebenläufigkeitskontrolle beim synchronen Editieren. Wenn bei den oben beschriebenen Verfahren Zugriff eher auf das ganze Dokument oder seine Teile erfolgt, wird bei den Transformationsverfahren auf kleinere Einheiten zugegriffen: auf die einzelnen Wörter oder Zeichen. Diese Verfahren sind deswegen gut für das synchrone Editieren geeignet. Sie erfüllen alle wichtigen Anforderungen: Die Antwortzeiten sind sehr kurz, da die lokalen Operationen sofort ausgeführt werden; die Bewahrung der Kausalität wird durch ein Zeitstempel-Verfahren erreicht; die Konvergenz der Ergebnisse und Intentionserhaltung werden durch die Definition eines Undo-Do-Redo-Schemas erfüllt, das auf einer totalen Ordnungsrelation der Operationen und einem jeweils client-seitig geführten History-Buffer der ausgeführten Operationen basiert. Zusätzlich wird jede Operation vor der Ausführung nach Bedarf transformiert, um auf die durch andere Operationen hervorgerufenen Änderungen zu reagieren [Gerlicher, Yang]. Einer der bekannten Algorithmen für die operationalen Transformationen ist *distributed Operational Transformation* (dOPT) [Ellisa].

Neben den Verfahren ohne Votierung existieren zahlreiche Verfahren mit Votierung. Diese Verfahren lösen das Problem der Nebenläufigkeitskontrolle durch Abstimmungen zwischen den einzelnen Teilnehmern. Eine Einigung auf eine Kontroll-/Synchronisationsentscheidung wird durch die Mehrheit der Stimmen erreicht, die Teilnahme aller Gruppenmitglieder ist dabei nicht notwendig.

2.3.4 Architekturen für kollaborative Editiersysteme

Zwei grundlegende Typen von Architekturen für kollaborative Editiersysteme können unterschieden werden: zentralisierte und replizierte Systemarchitekturen. Bei einer zentralisierten Architektur werden gemeinschaftliche Dokumente auf einem oder mehreren zentralen Servern gespeichert. Die Server übernehmen, abgesehen von den Verwaltungs- und Authentifizierungsaufgaben, die Zugriffskontrolle und die Propagierung der an Dokumenten durch-

geführten Änderungen. Die einzelnen Clients greifen auf den Server zu, um ein Dokument zu bearbeiten, der Server besitzt dabei immer eine aktuelle, gültige Version des Dokumentes. Eine Variante der zentralen Architektur ist das so genannte Information Sharing [Schwabe]. Bei dieser Architektur besitzt jeder Client eine Kopie des zu bearbeitenden Dokumentes. Der Server ist für die Sequenzialisierung der Operationen und die Synchronisation der verschiedenen Versionen des Dokumentes verantwortlich.

Bei zentralen Architekturen ist die Datenkonsistenz leicht zu wahren. Andererseits zeichnet sie relativ schlechte Ansprechbarkeit (lange Antwort- und/oder Benachrichtigungszeiten) aus, da die Propagierung der Änderungen immer über den zentralen Server stattfindet. Der Server kann dabei zum Engpass werden, was die Verfügbarkeit der gemeinsamen Dokumente erheblich verschlechtert. Die zentralisierte Architektur eignet sich aus diesen Gründen eher schlecht für eine synchrone Kollaboration, sie kann dagegen in den asynchronen Editiersystemen eingesetzt werden.

Bei einer replizierten Systemarchitektur hat jeder Teilnehmer eine Kopie des zu bearbeitenden Dokumentes. Lokale Operationen werden sofort ausgeführt und danach an andere Teilnehmer propagiert, ohne eine zentrale Stelle dazu einzubeziehen. Die Zugriffskontrolle und die Synchronisation der Operationen werden ebenfalls von jedem beteiligten Teilnehmer übernommen. Solch eine Architektur ermöglicht hohe Ansprechbarkeit und ist deswegen besonders für das synchrone kollaborative Schreiben attraktiv. Allerdings sind bei einer replizierten Architektur Daten-Inkonsistenzen nur schwer zu verhindern. Es können Probleme entstehen, z.B. beim Eintreten eines neuen Teilnehmers in die Gruppe. In solchem Fall muss sichergestellt werden, dass das neue Gruppenmitglied eine aktuelle Kopie des Dokumentes erhält, was bei einer streng replizierten Architektur mit einem gewissen Aufwand verbunden ist. Darüber hinaus liegt ein weiterer Nachteil von solchen Systemen im hohen Kommunikationsaufwand zwischen den Beteiligten.

Es existiert auch ein weiterer Architektur-Ansatz, in dem beide beschriebenen Architektur-Typen miteinander kombiniert werden, die so genannte hybride Systemarchitektur [Gerlicher]. Bei dieser Architektur besitzen sowohl der Server als auch einzelne Clients jeweils eine Kopie des gemeinsam genutzten Dokumentes. Die Clients propagieren ihre Änderungen an die anderen und an den zentralen Server. Somit liegt auf dem Server immer die aktuellste Version des Dokumentes. Diese Kopie steht für neue Teilnehmer zum Abruf bereit und kann unter anderem nach Bedarf zur Synchronisation genutzt werden. Mit solcher Architektur wird ein Versuch unternommen, die

Vorteile der beiden anderen Architekturen auszunutzen und so ihre Nachteile zu kompensieren.

Bei einer spontanen und/oder einer mobilen Kollaboration kann eine zentralisierte oder auch hybride Architektur von Nachteil sein. Die einzelnen Teilnehmer sind bei einer solchen Art der gemeinsamen Arbeit extrem autonom und die Gruppen-Beschaffenheit ist sehr dynamisch: Die Gruppenmitglieder können aufgrund ihrer Mobilität jederzeit das System betreten bzw. verlassen. Das Vorschreiben einer Verbindung zu einem zentralen Server kann zu einem Engpass führen, da unter den Bedingungen der Mobilität nicht immer von einer Internet-Verbindung ausgegangen werden kann. Eine spontane Adhoc-Kommunikation zwischen zwei oder mehreren Teilnehmern ist dagegen sehr wahrscheinlich. Eine Systemarchitektur, in der Gruppenmitglieder im Stande sind, sich selbst zu organisieren und sowohl in Rolle des Clients als auch des Servers aufzutreten, ist dagegen für den Kollaborationsprozess unter solchen Bedingungen von Vorteil. Solche Architekturen werden auch als Peer-to-Peer-Architekturen bezeichnet. Im nächsten Abschnitt werden der Begriff der Peer-to-Peer-Architektur und ihre wichtigsten Eigenschaften erläutert. Darüber hinaus werden bekannte Peer-to-Peer-Plattformen vorgestellt.

Peer-to-Peer-Architekturen

Eine Peer-to-Peer-Architektur zeichnen folgende Eigenschaften aus (nach [Dustdar]):

- Es gibt keine zentrale Koordination.
- Es gibt keine zentrale Datenbasis. Jeder Peer besitzt einen Teil des gesamten Datenbestandes.
- Die einzelnen Peers haben keine globale Sicht auf das System.
- Das globale Verhalten des Systems entsteht durch die Interaktionen der einzelnen Peers.
- Peers sind autonom.
- Peers und Verbindungen zwischen ihnen sind nicht notwendigerweise verlässlich.
- Die gesamten im System gespeicherten Daten müssen verfügbar sein, trotz verteilter Speicherung, möglicher Verbindungsausfälle etc.

Der Peer-to-Peer-Ansatz wird auf verschiedenen Abstraktionsebenen angewendet. Diesbezüglich können vier Abstraktionsebenen unterschieden werden: Netzwerkebene, Datenzugriffsebene, Dienstebene und Benutzerebene [Dustdar]. Aus der Sicht der kollaborativen Gruppe wird der P2P-Ansatz auf der Benutzerebene angewendet: Gruppierung von Teilnehmern und Unterstützung der Interaktionen zwischen ihnen.

Als weiteres Merkmal einer Peer-to-Peer-Architektur kann die Ausprägung des Hierarchie-Grades genannt werden. Es existieren folgende Modelle dazu:

- Zentralisiertes Modell: Es existiert eine zentrale Stelle, an der der globale Index aller Peers verwaltet wird. Diese Stelle übernimmt die Koordination der Peers.
- Dezentrales Modell: Es existiert kein globaler Index. Peers interagieren direkt miteinander.
- Hierarchisches Modell: Dieses Modell ist ein Zusammenspiel der beiden anderen Modelle. Es existieren normale Peers und hierarchische Super-Peers, die den globalen Index verwalten.

In weiteren Abschnitten werden besonders bekannte Peer-to-Peer-Plattformen vorgestellt und ihre wesentliche Eigenschaften erläutert.

ZeroConf Die ZeroConf-Spezifikation [ZeroConf] zeichnen vor allem drei folgende Funktionalitäten aus:

- Automatische Zuweisung der IP-Adressen ohne einen DHCP-Server;
- Automatisches Auffinden und Identifizierung von Diensten ohne einen zentralen Directory-Server;
- Übersetzung der Namen in IP-Adressen ohne einen DNS-Server.

Über ZeroConf können sich Geräte in einem IP-Netzwerk selbstständig konfigurieren und ihre Dienste sind ohne Kenntnis ihrer IP-Adressen oder Namen zu finden [Zivadinovic]. ZeroConf ermöglicht also die Veröffentlichung von Diensten und ihre Nutzung in einem Netzwerk dezentral. Es bietet damit die Grundlage für eine Peer-to-Peer-Architektur. Aber obwohl die Möglichkeit besteht, mit ZeroConf über Tunneling auf Dienste eines fremden Netzes zuzugreifen [Zivadinovic], ist dieses System in erster Linie für lokale Netzwerke

gedacht. Darüber hinaus ist ZeroConf nicht für Mobilität konzipiert: Es behandelt nicht das Monitoring und die Steuerung der Ressourcen in mobilen drahtlosen Netzwerken [Caituiro-Monge].

Des Weiteren kann ZeroConf den Peer-to-Peer-Ansätzen auf Netzwerk- und Datenzugriffsebene (siehe vorherigen Abschnitt) zugeordnet werden und bietet demzufolge keine Dienste für die Gruppenbildung und den Nachrichtenaustausch, die für die Ziele dieser Masterarbeit von Bedeutung sind. Es ist eher für die Verwaltung von Geräte-Diensten spezifiziert, wobei die Betonung auf Geräten liegt. Im Fall des Einsatzes von ZeroConf müssten deswegen dieser Dienst und andere ähnliche Services eigenständig implementiert werden.

Als eine bekannte Implementierung der ZeroConf-Spezifikation kann Bonjour von Apple genannt werden.

MS Windows Peer-to-Peer-Komponente Die Windows Peer-to-Peer Netzwerkkomponente [MSP2P] bietet eine Infrastruktur für Peer-to-Peer-Anwendungen unter Windows-Betriebssystemen. Für die Entwicklung von eigenen Applikationen wird von Microsoft das P2P SDK zur Verfügung gestellt, das sich nahtlos in die .NET-Entwicklungsumgebung integrieren lässt. Zu den weiteren Vorteilen kann die leichte Installation der Komponente gezählt werden.

Als Nachteil kann die Begrenzung der Nutzung dieser Komponente auf das Windows Betriebssystem und die .NET-Entwicklungsumgebung angesehen werden, sie ist also nicht plattformunabhängig. Außerdem kann keine Hand-Held-Device-Anwendung unter dieser Plattform entwickelt werden, da die mobilen Versionen von Windows diese Funktionalität nicht unterstützen. Darüber hinaus bringt diese Plattform die gleichen Einbußen wie jede kommerzielle Software: Keinen freien Zugriff auf den Source-Code der Komponente und somit keine Möglichkeit, die Funktionalität an eigene Bedürfnisse anzupassen und eventuelle Mängel nachzubessern.

JXTA JXTA [JXTA] ist ein Open-Source-Projekt, das von Sun ins Leben gerufen wurde. JXTA ist zurzeit das am meisten ausgereifte zur Verfügung stehende P2P-Framework [Shi]. Es standardisiert sechs Protokolle für den Aufbau und das Erhalten von spontanen Peer-to-Peer-Netzwerken, die die Suche nach anderen Peers, Bekanntmachung von und Suche nach Diensten, Erstellung von Peer-Gruppen, Kommunikation zwischen Peers und gegenseitige Überwachung ermöglichen. Es verspricht Interoperabilität zwischen unterschiedlichen Peer-to-Peer-Systemen und Plattformunabhängigkeit in Bezug auf Implementierungssprachen und Betriebssysteme. JXTA liegt ein hier-

archisches Modell zugrunde (siehe Abschnitt 2.3.4): Abgesehen von einfachen Peers existieren in JXTA auch so genannte Rendezvous- und Relay-Peers, zu deren Aufgaben die Weiterleitung der Anfragen und Nachrichten der anderen Peers gehört.

Die JXTA-Architektur besteht aus drei Schichten (siehe Abbildung 2.4). Den

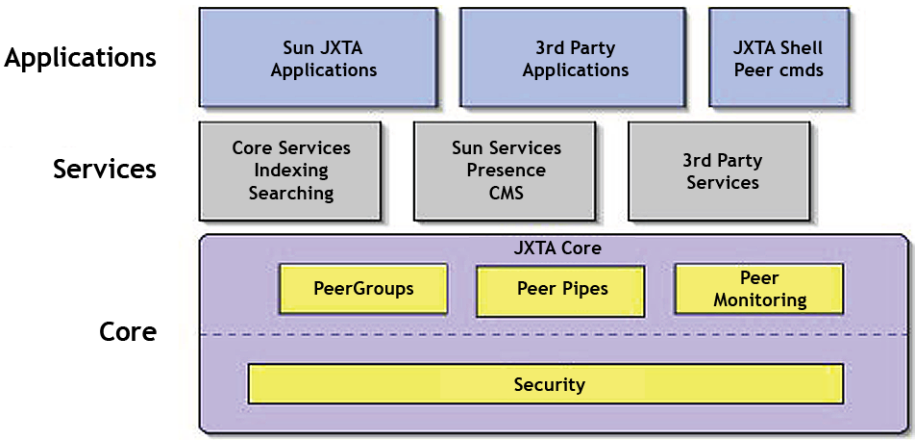

Abbildung 2.4: JXTA Architektur [JxtaProgGuide]

Kern bildet die Plattformschicht (*Core*). Hier sind die grundlegenden Funktionalitäten angesiedelt, die für ein Peer-to-Peer-Netzwerk üblich sind. Dazu gehören Mechanismen für die Entdeckung von Peers und Services und für die Kommunikation und Sicherheit. Diese Schicht ist unter anderem für die Steuerung der Gruppenbildung und das Monitoring verantwortlich. Die Dienstschicht (*Services*) unterstützt solche Dienste, die in einem Peer-to-Peer-Netzwerk nicht unbedingt notwendig, aber wünschenswert sind. Dazu gehören unter anderem Suche und Indizierung, File-Sharing und Authentifizierung. Diese Schicht erlaubt eine plattformunabhängige Kollaboration [Maibaum]. Die dritte Schicht (*Applications*) beinhaltet verschiedene integrierte Applikationen, z.B. für P2P Instant Messaging, Content Management, P2P E-Mail-Systeme usw. [JxtaProgGuide]. Diese Schicht benutzt die Funktionalität der Service- und Core-Schicht.

Es existieren bereits mehrere JXTA-Implementierungen sowohl in C als auch in Java. Außerdem wurde eine „Light“-Version für mobile J2ME-fähige Kleingeräte herausgebracht: JXTA Java Micro Edition [JXME]. Ein reines Hand-Held-Devices-Netzwerk über JXME ist allerdings nicht möglich, da JXME-Peers keine Services veröffentlichen können. Dafür ist eine permanente Ver-

bindung zu speziellen JXTA-basierten Peers nötig, den so genannten Relays [Bisignano].

2.3.5 Speicherungsart

Es kommen zwei Möglichkeiten für die Speicherung der Dokumente in Frage. Sehr verbreitet ist der Dateisystem-basierte Ansatz. Bei diesem Ansatz werden Dokumente, wie der Name schon sagt, in einem Dateisystem abgespeichert, entweder auf dem zentralen Server oder repliziert auf den Clients.

Bei einem Datenbank-basierten Konzept werden Dokumente in einem Datenbankmanagementsystem (DBMS) persistent gemacht und verwaltet. Dafür existieren unterschiedliche Ansätze. Die konventionelle Datenbank-basierte Methode beinhaltet die Speicherung des Dokumentes als einen Wert eines Attributes einer Relation. Es gibt aber auch andere Lösungen. Für XML-Dokumente existieren z.B. bereits spezielle XML-DBMS [Wong], die die Struktur der Dokumente abbilden und damit einen effizienten Zugriff auf einzelne Teile der Dokumente ermöglichen. Des Weiteren wurde im Rahmen eines Forschungsprojektes ein Datenbanksystem entwickelt, das die Dokumente Zeichen-basiert speichert [Leone] (mehr dazu im weiteren Abschnitt „Aktuelle Forschungsprojekte“).

Abgesehen von reinen Datei-basierten und Datenbank-basierten Speicherungsmethoden existiert auch ein Ansatz, der diese zwei Vorgehensweisen kombiniert. Diese Methode ist besonders in Dokumentenmanagementsystemen sehr verbreitet [Kampffmeyer]. Dabei werden die eigentlichen Dokumente auf einem oder mehreren Dateiservern gespeichert und ihre Meta-Daten in einem Datenbanksystem. Diese Daten beinhalten, abgesehen von trivialen Informationen wie dem Namen und dem Erstellungsdatum, auch Charakteristika, die eine Aussage über den Inhalt des Dokumentes machen, z.B. Schlagwörter. Die Führung und Pflege von diesen Eigenschaften sind maßgebend für eine effiziente Suche nach benötigten Dokumenten in solchem System.

2.4 Replikation der Dokumente

Wie schon in den vorherigen Abschnitten erläutert wurde, hat eine replizierte Systemarchitektur mehr Potenzial für eine spontane kollaborative Dokumentenbearbeitung in mobilen Umgebungen. Bei diesem Ansatz hat jeder Teilnehmer eine lokale Kopie des zu bearbeitenden Dokumentes. Dadurch können

unterschiedliche Versionen eines Dokumentes bei den einzelnen Teilnehmern entstehen, da keine permanente Netzwerk-Verbindung zwischen ihnen vorausgesetzt werden kann und die Änderungen nicht an alle sofort propagiert werden können. In diesem Zusammenhang sind folgende Fragestellungen von Bedeutung:

- Wie wird ein konsistenter Zustand des Dokumentes erreicht und wo?

- Bekommen die einzelnen Peers irgendwann einen gleichen Zustand des Dokumentes?

Um diese Probleme zu lösen, muss eine passende Replikationsstrategie überlegt werden. Da die Teilnehmer nicht immer online sind, können nur optimistische Verfahren verwendet werden, die eine schwache Konsistenz mit verzögertem Aktualisieren erlauben. Bei diesen Verfahren stellen die Replikate nur eine Momentaufnahme der primären Daten zu einem bestimmten Zeitpunkt dar. Eine mögliche Variante solcher asynchronen Replikation ist die so genannte Merge-Replikation. Darunter werden nicht synchronisierte Änderungen eines replizierten Dokumentes an mehreren gleichberechtigten Knoten und asynchrone Propagierung dieser Änderungen verstanden [Härder]. Das Problem bei dieser Art der Replikation liegt in Konflikten, die bei der Zusammenführung der unterschiedlichen Versionen entstehen können.

Eine Alternative bietet eine andere Strategie der asynchronen Replikation: Ein Primär-Kopie-Verfahren, bei dem eine Hauptinstanz des Dokumentes existiert und jeweils eine Kopie bei jedem beteiligten Gruppenmitglied. Die Änderungen werden in der Hauptinstanz zusammengeführt. Das Problem hier stellt das Risiko eines Ausfalls des Knotens mit der primären Kopie des Dokumentes dar. In dieser Situation sollte eine neue Hauptinstanz gewählt werden, um den Arbeitsfluss nicht zu stören, z.B. mit Hilfe eines Wahl-Algorithmus. Da aber kein permanentes Netz vorhanden ist, kann nicht sichergestellt werden, ob der ursprüngliche Primär-Kopie-Knoten wirklich ausgefallen ist oder nur längere Zeit keine Verbindung hat. Das kann zum Problem führen, dass mehrere Primär-Kopien im System entstehen werden. Dementsprechend sollte eine Strategie zur Lösung dieses Problems im System existieren.

Unabhängig davon, welches Verfahren für die Replikation gewählt wird, bleibt die Problematik des Merging von unterschiedlichen Versionen eines Dokumentes bestehen. Als Lösung kann der Versuch angesehen werden, das Entstehen von voneinander abweichenden Zuständen gar nicht zuzulassen. Das kann z.B. durch die Vergabe von exklusiven Editierrechten auf die einzelnen Teile des Dokumentes erreicht werden. Dieses Konzept kann auch als eine Variante der dauerhaften Sperren betrachtet werden. Dadurch kann ein

Dokumententeil nur von einem Teammitglied zu einem Zeitpunkt bearbeitet werden. So können keine Inkonsistenzen entstehen und das Merging-Problem wird auf das Zusammenführen der einzelnen Teildokumente minimiert.

Als eine Verstärkung der Kollaboration könnte das Anbringen von Annotationen durch ein Teammitglied an „fremde“ Dokumententeile, auf die es keine Editierrechte besitzt, angesehen werden. Diese Maßnahme kann zur Verbesserung der Dokumentenqualität und Steigerung der Kooperationsunterstützung führen.

Bei dem Sperren-Konzept taucht jedoch ein anderes Problem auf: Was passiert, wenn ein Gruppenmitglied, das ein exklusives Recht auf einen Teil des Dokumentes besitzt, relativ lange nicht erreichbar ist? Die kollaborative Bearbeitung des Dokumentes sollte nicht durch ein solches Ereignis unterbrochen werden. Beim „Verschwinden“ eines Beteiligten sollte deswegen gewöhnlich ein anderes Gruppenmitglied seine Aufgabe übernehmen. Das System muss folglich mit solchen Problemen umgehen können, z.B. durch die Möglichkeit, Editierrechte aufzuheben.

2.5 Aktuelle Forschungsprojekte

Besonders in Forschungsprojekten für mobile Umgebungen werden verschiedene vorgestellte Ansätze miteinander kombiniert, um möglichst alle ihre Vorteile auszunutzen und so die besten Ergebnisse zu erzielen. In weiteren Abschnitten werden an Beispielen von einigen aktuellen Projekten diese verschiedene Design-Ansätze vorgestellt.

2.5.1 TeNDaX

Text Native Database Extension (TeNDaX) [Leone, Hodel] ist ein Forschungsprojekt an der Universität Zürich, dessen Kern die Entwicklung eines Konzeptes für pervasive Echtzeit-Editier- und Managementsysteme bildet und Dokumente kollaborativ, überall und mit beliebigen Geräten zu editieren ermöglicht (nach [Leone]). Das Projekt geht weg von dem etablierten Ansatz der Operationalen Transformationen und stellt eine andere interessante Idee für synchrones Bearbeiten von Dokumenten vor. Dem Konzept liegen folgende Prinzipien zugrunde:

- Das Konzept basiert auf Datenbank-Verfahren: Die Dokumente werden in einer nativen Form in der Datenbank gespeichert.

- Die Clients haben keine Repliken der Dokumente im Sinne von Datenbanken-Repliken, sondern so genannte „Images“ auf das Dokument. Damit existiert ein Dokument nur einmal im System und mehrere Autoren arbeiten gleichzeitig auf ein und derselben Dokument-Instanz.
- Das Editieren wird als Echtzeit-Transaktionen angesehen und behandelt. Die Konkurrenzkontrolle wird also automatisch vom DBMS übernommen.
- Jedes Zeichen und Symbol eines Dokumentes wird als ein Objekt repräsentiert und separat in der Datenbank gespeichert. Die Änderungs-Transaktionen sind damit Zeichen-basiert, folglich sehr kurz, was kurze Latenzzeiten verspricht.
- Die Text-Zeichen werden in der Datenbank in Form von doppelt-verketteten Listen gespeichert, um die Reihenfolge der einzelnen Characters eines Dokumentes zu erhalten und die Änderungsoperationen an den einzelnen Zeichen (delete, insert, update) nachvollziehbar zu machen.
- Es wird eine strikte Trennung zwischen dem Inhalt des Dokumentes und seinen Layout-Informationen durchgezogen. Für ein Dokument können mehrere Layouts existieren, was die Adaptivität der Dokumenten-Darstellung an die bei dem Editieren benutzten Geräte verspricht.

Architektur

Die Abbildung 2.5 zeigt eine Skizze der TeNDaX-Architektur. Wie in der Abbildung zu sehen ist, besteht die Architektur aus drei Schichten: Datenschicht, Business-Logik und Präsentationsschicht. Die Datenschicht bilden eine oder mehrere Datenbanken, in denen Dokumente gespeichert werden. Unter der Präsentationsschicht werden Text-Editoren verstanden, die auf den einzelnen Clients laufen. Die Business-Logik beinhaltet den zum kollaborativen Editieren benötigten Funktionalitäts-Umfang des Systems. Diese Schicht besteht aus drei Komponenten. Ein oder mehrere Applikations-Server (kurz AS) ermöglichen das Editieren von Dokumenten innerhalb einer Datenbank-Umgebung und sind verantwortlich für Awareness, Security, Dokumentenmanagement etc. Die Real-Time-Server-Komponente (kurz RTSC) ist für die Sofort-Propagierung jeder an einem Client ausgeführten Operation an alle Client-Editors verantwortlich. Der Web-Server stellt web-basierte Funktionalitäten zur Verfügung wie das Einloggen der User oder die Suche und das Öffnen von Dokumenten, aber keine Editiermöglichkeiten.

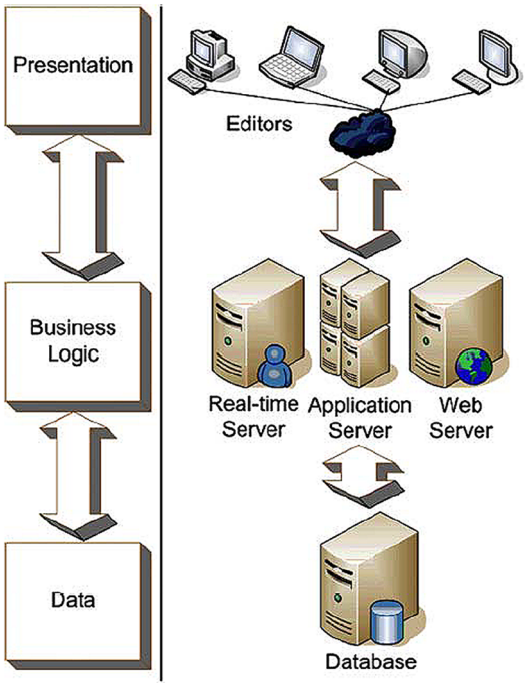

Abbildung 2.5: TeNDaX: Architektur-Skizze [Hodel]

Ablauf

In der Abbildung 2.6 ist ein beispielhafter Ablauf skizziert: Bei einer auf einem Client durchgeführten Änderungsoperation wird diese an den Applikations-Server gesendet. Dieser seinerseits leitet die Änderung an die Datenbank weiter, wo die Operation ausgeführt wird. Wenn die Transaktion erfolgreich abgeschlossen wurde, wird die Änderung an den Real-Time-Server übergeben und von ihm an alle Clients propagiert, zu denen eine Verbindung existiert. In TeNDaX existieren zwei Verhaltensvarianten des Editors bei den Manipulationen an Dokumenten. Die erste ist ein strikt synchroner Ansatz, bei dem die durchgeführte Manipulation erst dann im Editor sichtbar wird, wenn die Datenbank-Transaktion erfolgreich war. Bei diesem Ansatz werden exklusive Sperren für die Konkurrenzkontrolle eingesetzt. Der andere Ansatz ist asynchron (im Kontext des Echtzeit-Editierens). Bei diesem Ansatz wird die Änderung am Dokument sofort im Editor sichtbar, danach wird sie in der Datenbank ausgeführt. Wenn die Transaktion erfolgreich war, wird die Änderung an alle anderen propagiert, den Initiator dieser Änderung ausgeschlossen (siehe Abbildung 2.6). Wenn nicht, dann muss sie beim Initiator

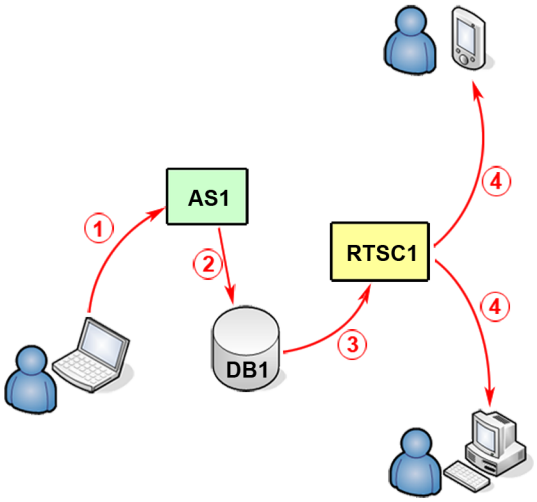

Abbildung 2.6: TeNDaX: Ablauf-Skizze, nach [Leone]

rückgängig gemacht werden. Bei diesem Ansatz wird ein Validierungsverfahren für die Konfliktauflösung verwendet.

Annehmbarkeit für mobile Umgebungen

Als Ziel des Forschungsprojektes wurde das pervasive Bearbeiten von Dokumenten mit beliebigen Geräten gestellt. Allerdings haben Hand-Held-Geräte, wie PDA, eine kleinere Übertragungs-Bandbreite. Damit können größere Latenzzeiten entstehen, die sich negativ auf das Verhalten des Systems auswirken können. Dazu kommt zusätzlich, dass in mobilen drahtlosen Umgebungen nach dem aktuellen Stand der Technik ein permanenter Online-Betrieb unmöglich oder nur schwer einzuhalten ist, das Konzept von TeNDaX aber sieht keinen Offline-Editiermodus vor. Von anderer Seite ist dieser Ansatz sehr wohl in „geschlossenen“ lokalen Netzwerken funktionsfähig, wo von einem permanenten Netz ausgegangen werden kann, wie z.B. innerhalb eines Unternehmens.

2.5.2 MCWS

Wie im vorherigen Abschnitt zu sehen war, ist beim kollaborativen Schreiben in mobilen Umgebungen die Anpassungsfähigkeit der dafür benutzten Anwendungen an die Umgebungsbedingungen besonders wichtig. Darunter

wird die Flexibilität des Funktionsumfanges und des Verhaltens der Anwendung abhängig von der aktuellen Umgebung, wie z.B. die Existenz eines Netzwerk-Zuganges oder die Übertragungs-Bandbreite des zur Verfügung stehenden Gerätes, und die Anpassbarkeit der Darstellung an die Möglichkeiten des Endgerätes verstanden. Rein synchrones kollaboratives Schreiben unter dem Mobilitätsaspekt ist deswegen fast unmöglich. Für ein adaptives mobiles System scheint deshalb ein hybrides Konzept besonders interessant zu sein, in dem eine Möglichkeit sowohl für synchrones als auch für asynchrones Editieren gegeben wäre. In diesem Abschnitt wird ein Forschungsprojekt vorgestellt, das dieses Konzept umsetzt und so die höchstmögliche Adaptivität der Anwendungen zu erreichen versucht.

Mobile Collaboration Writing System (MCWS) [Yushun] ist ein Forschungsprojekt an der Tsinghua Universität, China. In diesem Projekt wurden ein mobiles kollaboratives System ausgearbeitet, basierend auf den Technologien für adaptive drahtlose Groupware [Yushun], und ein Proof-of-Concept-Prototyp entwickelt. Das System weist folgende grundlegende Eigenschaften auf:

- Strukturierte Co-Dokumente: Im Dokument werden Informationen über seine Struktur (Kapitel, Paragrafen usw.) gespeichert. Außerdem enthält das Dokument kollaborative Informationen wie Status, Autorisierung, Version etc. Darüber hinaus ist die Darstellung in einem beliebigen traditionellen Format möglich.

- Hybride Architektur: Zentralisierte Architektur wird für asynchrone Zugriffe eingesetzt, replizierte Architektur für synchrone Sessions.

- Netzwerk-Monitoring-Modul für die Adaptivität: Durch das Monitoring-Modul wird der Zustand des drahtlosen Netzwerkes überwacht. Größen wie Bandbreite, Fehlerrate, Signalstärke und Latenzzeit werden gemessen und ausgewertet, um darauf adäquat zu reagieren.

- Synchrone Awareness: Jedes Gruppenmitglied erhält einen Echtzeit-Status über andere kollaborierende Teilnehmer. Zur Konkurrenzkontrolle bei synchronen Sessions werden Operationale Transformationen eingesetzt. Die Erhaltung einer Session wird auch bei einem Verbindungsabbruch gewährleistet. Darüber hinaus wird die Verbindung automatisch hergestellt, sobald die Möglichkeit dazu besteht (self-healing).

- Asynchrone Awareness: Bei asynchronen Zugriffen werden Informationen über die Erreichbarkeit der anderen Teilnehmer, die kollaborative Historie und Lock-Informationen zur Verfügung gestellt.

Netzwerk-Modell

In der Abbildung 2.7 ist das Netzwerk-Modell des Systems dargestellt, das aus drei wesentlichen Komponenten besteht: drahtgebundenem Netzwerk, Mobilen Hosts (MH) und Mobilen Support-Stationen (MSS). Die MSS die-

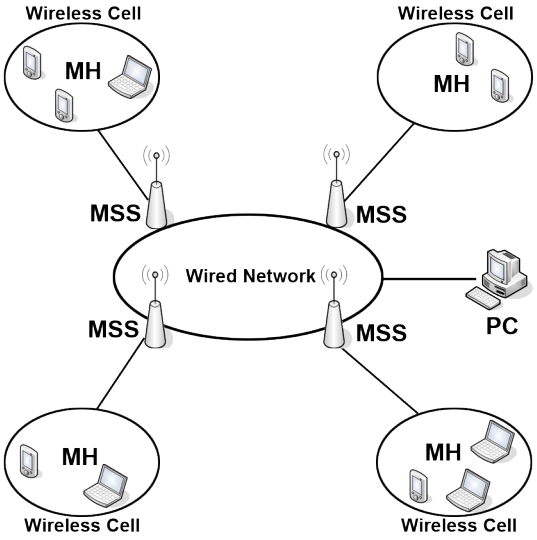

Abbildung 2.7: MCWS: Netzwerk-Modell (nach [Yushun])

nen als Access Points für mobile Hosts. Sie können Informationen und Teildokumente für ihre einzelnen MH's zwischenspeichern, wenn MH aktuell zu kleine Ressourcen haben. Beim Wechsel der Zelle eines mobilen Hosts wird der Austausch der MH-bezogenen Informationen zwischen seiner alten und der neuen MSS durchgeführt. Die Session-bezogenen Informationen werden auf den MSS und nicht auf den einzelnen MH's gehalten. Das ermöglicht kollaborative Arbeit der MH's ohne zusätzlichen Aufwand wie den Neuaufbau der Session.

Architektur

Die Architektur von MCWS besteht aus drei Teilen: dem *Mobilen Host*, dem *Access Server* und dem *Collaborative Writing Server* (siehe Abbildung 2.8). Der *Mobile Host* beinhaltet im Wesentlichen folgende Komponenten: Session-Module für die Unterstützung der synchronen und asynchronen Sessions, Awareness-Modul und Module für die Anpassbarkeit an die Umgebungsbedingungen.

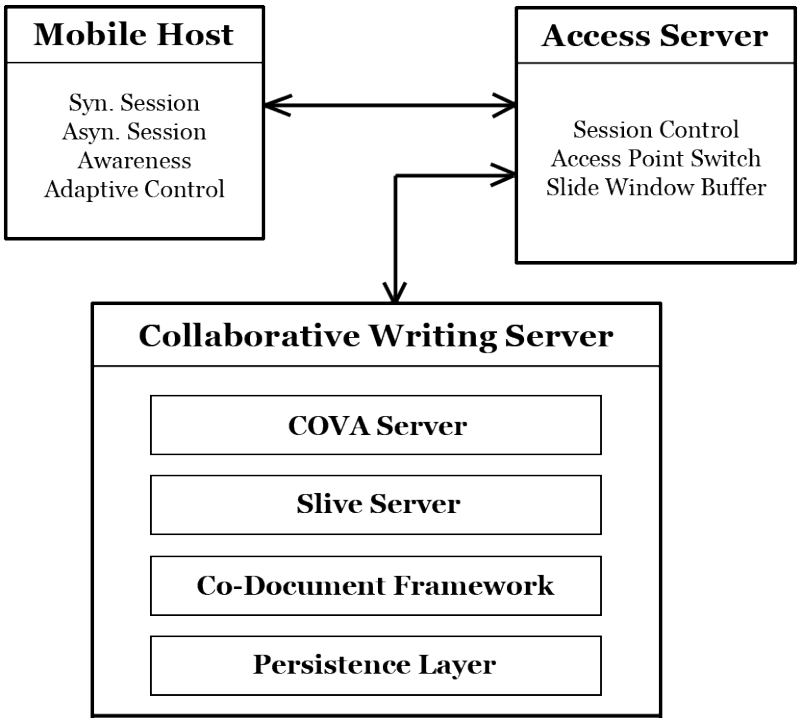

Abbildung 2.8: MCWS: Architektur-Skizze (nach [Yushun])

Der *Access Server* ist das Verbindungsglied zwischen den mobilen Hosts und dem *Collaborative Writing Server*. Er übernimmt die Überwachung des Session-Zustandes seiner einzelnen Hosts, die Wiederherstellung der Session beim Wiedereintreten des Hosts ins Netzwerk und die Weiterleitung der Session-Informationen an den anderen *Access Server* beim Wechsel des Access Points durch den mobilen Host (siehe vorherigen Abschnitt).

Der *Collaborative Writing Server* ist das Kernstück des Systems. Er beinhaltet Komponenten zur Unterstützung von synchronen Sessions (COVA Server und Slave Server), das Co-Dokumenten-Framework und die Persistenz-Schicht. Das synchrone Bearbeiten von Dokumenten wird über einen replizierten Ansatz umgesetzt: Der *Collaborative Writing Server* verteilt eine synchrone Session auf mehrere Slave Servers. Co-Dokumenten-Framework sorgt unter anderem für adaptive Darstellung der Dokumente auf den Hosts.

Annehmbarkeit für mobile Umgebungen

Durch eine hybride Architektur und die Unterstützung sowohl des synchronen als auch des asynchronen kollaborativen Arbeitens stellt das Konzept von MCWS eine gute Lösung für kollaboratives Schreiben in mobilen Umgebungen dar. Darüber hinaus erreicht MCWS einen hohen Grad von Adaptivität, was unter den Aspekten der Mobilität und heterogenen Geräten eine sehr wichtige Rolle spielt.

2.5.3 Weitere Forschungsarbeiten

In den folgenden Abschnitten werden einige weitere interessante Projekte im Bereich der kollaborativen Dokumenten- und Datenbearbeitung in mobilen Umgebungen kurz vorgestellt.

2.5.3.1 PASIR

PASIR [Neyem] ist eine Plattform für kollaborative Nutzung von Dokumenten in mobilen Ad-hoc-Netzwerken, speziell für Hand-Held-Geräte entwickelt, basierend auf XML. Diese Plattform erlaubt das Offline-Manipulieren von Dokumenten und die Online-Synchronisation von Repliken. Im Rahmen dieses Projektes wurde ein Abgleichsalgorithmus für XML-Bäume entwickelt, der drei wesentliche Teile aufweist: TreeDiff, TreeMerge und TreeReconcile (mehr dazu siehe in [Neyem]).

2.5.3.2 SODA

SODA [Wong] ist ein natives XML-Datenbankmanagementsystem, das an der Universität von New South Wales, Sydney, Australia entwickelt wurde. SODA bietet ein kollaboratives Editiersystem für mobile Geräte, das einen sehr effizienten Mechanismus zur Synchronisierung von Updates der konkurrierenden Editoren enthält. Innerhalb des Systems sind synchrone und asynchrone Sessions möglich. Kernpunkte des Designs sind die Nutzung der erweiterten XML Query Language (XQL), lokale Datenbanken auf den Clients und die Nutzung des Servers als einen Kanal für Updates bei einer vorhandenen Netzwerk-Verbindung.

2.5.3.3 XMIDDLE

XMIDDLE [Mascolo] ist eine Peer-to-Peer Middleware für mobile Ad-hoc-Netzwerke, die eine transparente Verteilung von XML-Dokumenten zwischen heterogenen mobilen Peers ermöglicht und Funktionalitäten zu Daten-Replikation und -Abgleich besitzt. Daten werden in XMIDDLE in Form von XML-basierten Bäumen repräsentiert. Das Middleware erlaubt sowohl einen synchronen Datenaustausch als auch ein asynchrones Offline-Editieren. Für die Nutzung von „fremden“ Daten wurde ein „Linking“-Konzept entwickelt, das vergleichbar mit dem Mounting-Prinzip von Netzwerk-Dateisystemen in verteilten Betriebssystemen ist.

2.5.3.4 TouchSync

TouchSync [Roussev] ist ein Framework für Ad-hoc-Kollaboration auf Hand-Held-Devices und wurde speziell für Palm-Geräte entwickelt. Das Framework besitzt einen flexiblen Sharing-Mechanismus, der ermöglicht, Single-User-Applikationen leicht für die Kollaboration anzupassen. Es ist Datenbank-basiert und hat drei wesentliche Komponenten: einen Session-Manager, ein Tool für Ad-hoc-Sharing für Datenbanken und einen Zugriffsmanager.

2.6 Resümee

In diesem Kapitel ist deutlich geworden, dass bei der Entwicklung eines Systems zur Bearbeitung von Dokumenten in einer Gruppe folgende Faktoren eine sehr wichtige Rolle spielen:

- *Wo werden sich die einzelnen Benutzer bei der gemeinsamen Arbeit befinden (in einem Raum, in einem Gebäude oder noch stärker verteilt)?* Bei großen räumlichen Verteilung der Teilnehmer muss das kollaborative System im größeren Maße die Arbeitsschritte der einzelnen Benutzer koordinieren als bei der Arbeit in einem Raum. Dementsprechend muss das System die Überwachung der Benutzer-Aktivitäten übernehmen und die Kommunikation zwischen ihnen stärker unterstützen. Bei der Arbeit in einem Raum kann die Koordination dagegen schwächer ausfallen, da die Gruppenmitglieder auch auf natürliche Weise miteinander kommunizieren und ihre Tätigkeiten aufeinander abstimmen können. Aus gleichem Grund hat das synchrone kollaborative Schreiben mehr Potenzial beim Arbeiten in einem Raum. Bei der örtlich getrennten

Arbeit kann synchrones Editieren dagegen zu gegenseitigen Störungen der Benutzer führen.

- *In welcher Umgebung soll das System eingesetzt werden (feste oder dynamische Netzwerk-Topologie)?*
  Der Grad der Mobilität der einzelnen Kollaborationsteilnehmer hat eine Auswirkung auf die Wahl der Systemarchitektur. Bei starker Mobilität kann nicht davon ausgegangen werden, dass die Benutzer immer eine Internet-Verbindung haben. Aus diesem Grund ist der Ansatz einer zentralisierten Architektur unter solchen Bedingungen eher von Nachteil. Bei einem hohen Grad der Mobilität kommt das reine synchrone Schreiben aus demselben Grund nicht in Frage.

- *Welcher Freiheitsgrad soll dem Benutzer in Bezug auf die Gestaltung seiner Arbeitsschritte zur Verfügung stehen (nur gleichzeitig mit anderen Gruppenmitgliedern oder auch offline arbeiten)?*
  Das Schreiben von Dokumenten ist ein sehr individueller und kreativer Prozess, abgesehen vielleicht von der Erstellung von technischen Dokumentationen und ähnlichen Schriftstücken. Das Schreiben erfolgt gewöhnlich zu verschiedenen Tageszeiten und wird in Abhängigkeit von der Persönlichkeit des Verfassers unterschiedlich gestaltet. Ein kollaboratives System für Dokumentbearbeitung kann deswegen dem Benutzer nicht vorschreiben, seine Dokumente oder Teile eines gemeinsamen Dokumentes nur gleichzeitig mit anderen Gruppenmitgliedern zu editieren.

- *Welche Art der Kollaboration wird angestrebt (spontane und für relativ kurze Zeit oder geplante für langfristigere Projekte)?*
  Für spontane Zusammenarbeit hat ein dezentraler und replizierter Systemaufbau ein größeres Potenzial als bei einem zentralisierten System, da die Spontanität leichtgewichtige Lösungen erfordert, die auch ohne Internet-Verbindung auskommen können und kein aufwändiges Aufsetzen des Systems beanspruchen. Für langfristige Projekte lohnt sich dagegen der Einsatz einer zentralisierten Lösung mit dem damit verbundenen organisatorischen und zeitlichen Aufwand und den zusätzlichen Ressourcen-Kosten, da die Verwaltung der gemeinsamen Arbeitsfortschritte in einem solchen System leichter ist.

Kapitel 3

Analyse

In diesem Kapitel wird die Vision dieser Masterarbeit vorgestellt. Im Weiteren werden am Beispiel eines Szenarios die damit verbundenen Ziele und Aufgaben präsentiert. Dazu werden wichtigste Anwendungsfälle anhand des Szenarios konzipiert und die Funktionalität des zu entwickelnden Systems definiert.

3.1 Vision

Es gibt viele Situationen für spontane Kollaboration. Zu solcher Kollaboration gehört unter anderem auch die spontane Bildung einer Arbeitsgruppe für die Lösung einer relativ kurzfristigen Aufgabe, einschließlich das Verfassen eines gemeinsamen Dokumentes. Bei solchen Aufgaben ist oft eine leichtgewichtige Unterstützung der Zusammenarbeit erwünscht, die einfach und vor allem schnell die Gruppenmitglieder zu der eigentlichen Arbeit übergehen lässt und die einzelnen Teilnehmer nicht in ihrer Mobilität einschränkt.

Zwei Aspekte sind tragend für diesen Kollaborationstyp: Dokument-Bearbeitung und spontane Zusammenarbeit.

Das typische Verfassen eines Dokumentes durch mehrere Personen erfolgt in den meisten Fällen nach dem Prinzip der Aufteilung der Aufgaben. So werden z.B. im Dokument mehrere inhaltlich in sich geschlossene Teile bestimmt und jeder Beteiligte übernimmt das Schreiben eines der Teilen. Das gleichzeitige Bearbeiten desselben Textabschnitts durch mehrere Personen ist dagegen in der Praxis eher unüblich. Um sich gegenseitig über den aktuellen Stand des gemeinsamen Dokumentes zu informieren, tauschen die Beteiligten untereinander ihre Fassungen des Dokumentes aus. Das Erstellen der gemeinsamen

Dokument-Version erfolgt durch das (unter Umständen mühsame) Zusammenführen der einzelnen Versionen des Dokumentes, typisch durch das Kopieren/Einfügen der entsprechenden Abschnitte. Dabei können Brüche in der Formatierung entstehen (andere Schriftart, Schriftgröße usw.), die auch zu bewältigen sind. Dazu kommen gegenseitige Korrekturen des Textes, sowohl inhaltliche als auch der Rechtschreibung. Dieser Aspekt führt zu noch mehr Aufwand bei der Erstellung einer gemeinsamen Fassung des Dokumentes. Generell ist der Gesamtprozess mit einem großen Datenverkehr und einem organisatorischen Kommunikationsaufwand verbunden.

Eine spontane Zusammenarbeit zeichnet sich vor allem durch das Anstreben aus, so schnell wie möglich mit der Arbeit anfangen zu können. Ein unkompliziertes und schnelles Aufsetzen der nötigen Werkzeuge ist dabei wünschenswert. Ein großer Installations-Aufwand ist dagegen eher nachteilig. Die Spontanität wird darüber hinaus dadurch charakterisiert, dass die Zusammenarbeit überall dort stattfinden kann, wo sich mehrere Personen mit gleichen Zielen treffen. Dementsprechend ist das Kollaborationsumfeld mobil und sehr dynamisch. Außerdem stehen den Beteiligten nur eingeschränkte Ressourcen zur Verfügung.

Die Idee dieser Masterarbeit liegt darin, ein System zu entwickeln, das die spontane kollaborative Bearbeitung von Dokumenten unterstützen wird. Das System wird demnach die Organisation der Beteiligten und die Zusammenführung der verschiedenen Dokument-Versionen übernehmen, den Benutzern das Anbringen von gegenseitigen Korrekturen und Ergänzungen erleichtern und in dynamischen und mobilen Umgebungen einsetzbar sein. Darüber hinaus wird angestrebt, die gemeinsame Bearbeitung von Dokumenten in einer Arbeitsgruppe zur jeder Zeit und an jedem Ort zu ermöglichen, also auch offline.

Das System wird aus diesem Grund ein asynchrones kollaboratives Schreiben unterstützen. Das Aufsetzen eines eigenen Servers für die Verwaltung der Arbeitsschritte ist dafür ungeeignet, da dies einen zusätzlichen Aufwand erfordert, was bei einer spontanen Kollaboration, wie es schon beschrieben wurde, eher nachteilig ist. Vor allem aber macht diese Lösung die gemeinsame Arbeit von einer zentralen Stelle abhängig, was bei einer solchen mobilen Art der Zusammenarbeit nicht akzeptabel ist. Wenn eine hohe Flexibilität erwünscht ist, ist eine dezentrale Lösung von Vorteil. Aus diesem Grund wird für das System ein dezentraler Ansatz gewählt, der die Koordination der Gruppenmitglieder basierend auf einer Peer-to-Peer-Kommunikation unterstützt. Die Beteiligten können bei diesem Ansatz sowohl online als auch ad hoc miteinander arbeiten.

Die Arbeit an gemeinsamen Objekten kann in den meisten kollaborativen Systemen zu der Nebenläufigkeit der Zugriffe auf diese Objekte führen. Die Nebenläufigkeit bei der kollaborativen Arbeit hat sowohl Vorteile als auch Nachteile. Zu den Vorteilen kann der hohe Grad des kollaborativen Aspekts gezählt werden. Nachteilig dagegen ist die Notwendigkeit der Anwendung von komplexen Verfahren zu ihrer Kontrolle (siehe Abschnitt 2.3.3). Gestützt auf die oben beschriebene Vorstellung vom gemeinsamen Verfassen eines Dokumentes wird in dieser Masterarbeit ein Konzept entwickelt, das der Nebenläufigkeit durch eine feine Strukturierung des gemeinsamen Objektes (des Dokumentes) und eine präzise Verteilung der Aufgaben vorbeugt. Das gemeinsame Dokument wird eine hierarchische, verschachtelte Struktur mit einer uneingeschränkten Tiefe haben können (siehe Abbildung 3.1). Jeder

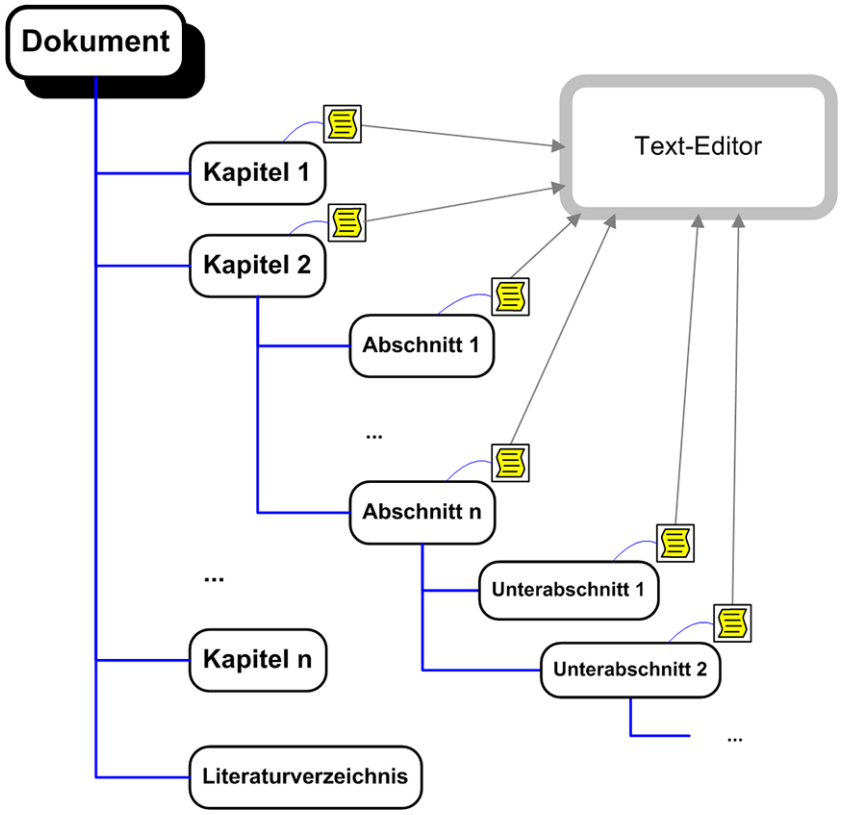

Abbildung 3.1: Hierarchische Dokumentenstruktur

Knoten dieser Struktur wird ein unabhängiges Teildokument darstellen, das von einer der beteiligten Personen bearbeitet werden kann. Alle Teildoku-

mente werden aber das gemeinsame Dokument bilden (mehr dazu in weiteren Abschnitten).

Für das eigentliche Schreiben an Dokumenten wird in diesem kollaborativen System ein existierendes Textbearbeitungsprogramm eingesetzt. Durch die Trennung des Inhaltes des Dokumentes von seiner logischen Struktur und weiteren Daten (mehr dazu in dem Entwurf-Kapitel) wird den Benutzern die Möglichkeit gegeben, ihre Dokumententeile in einem beliebigen Editor zu editieren, der das Format des Dokumentes unterstützt. In Abbildung 3.1 sind die Inhalte des Dokumentes als kleine Rechtecke neben den einzelnen Knoten dargestellt. Dieser Editor wird dem Benutzer vom System nicht vorgeschrieben, sondern ist von ihm frei wählbar.

3.2 Aufgabenstellung

Am folgenden Beispielszenario soll die Idee der Masterarbeit anschaulich präsentiert werden. Des Weiteren werden anhand dieses Szenarios die wichtigsten Anwendungsfälle für das zu entwickelnde System abgeleitet und die System-Funktionalität definiert.

3.2.1 Beispielszenario

Als Grundlage für die Idee der Masterarbeit wurde folgendes Beispielszenario genommen.

Eine Studentengruppe will eine Seminar-Ausarbeitung zum Thema „Klassifizierung der kollaborativen Systeme“ gemeinsam verfassen. Die Zusammensetzung der Gruppe sowie die Entscheidung zur Zusammenarbeit fällt dabei spontan, bei einem Treffen. Daran wollen sich drei Studenten beteiligten: *Jan*, *Lisa* und *Maik*. Die Beteiligten wollen für eine produktivere Arbeit Aufgaben untereinander aufteilen, damit jeder von ihnen nur einen Teil des Dokumentes zu Bearbeitung bekommt und das Erstellen des ganzen Dokumentes schneller vorangeht. Darüber hinaus wollen die Teilnehmer bei ihrer Arbeit keine räumlichen Einschränkungen haben und ihre Arbeit immer dann fortsetzen, wenn sie gerade eine Gelegenheit dazu haben, z.B. unterwegs mit einem Notebook, bei sich zu Hause oder auch in der Hochschule. Die Abhängigkeit voneinander ist dabei auch unerwünscht, da die Studenten an ihrem Dokument nicht nur bei den gemeinsamen Treffen (realen oder virtuellen) arbeiten wollen, sondern auch unabhängig von den anderen. Der Abgleich der schon

durchgeführten Arbeit soll aber trotzdem gelegentlich stattfinden, damit die Teilnehmer über die gemeinsamen Fortschritte informiert sind.

Das Szenario läuft folgendermaßen ab. Als Erstes erstellen die Studenten eine neue Gruppe im System: ihre Arbeitsgruppe „Seminar“. Danach erstellen sie das zu bearbeitende Dokument und legen seine vorläufige Struktur fest. Diese Aufgabe wird durch einen der Studenten übernommen, den *Jan*. Im weiteren Verlauf des Szenarios wird dieser Teilnehmer als „Initiator“ auftreten. Nachdem das Dokument erstellt wurde, verteilt der Initiator die Aufgaben an die einzelnen Beteiligten: Er weist jeweils einen oder mehrere Abschnitte des Dokumentes dem Teilnehmer zu, der diesen Abschnitt bearbeiten will. In Abbildung 3.2 ist die Dokumentenstruktur mit den Namen der den einzelnen Abschnitten zugewiesenen Studenten nach der Start-Phase der kollaborativen Arbeit dargestellt.

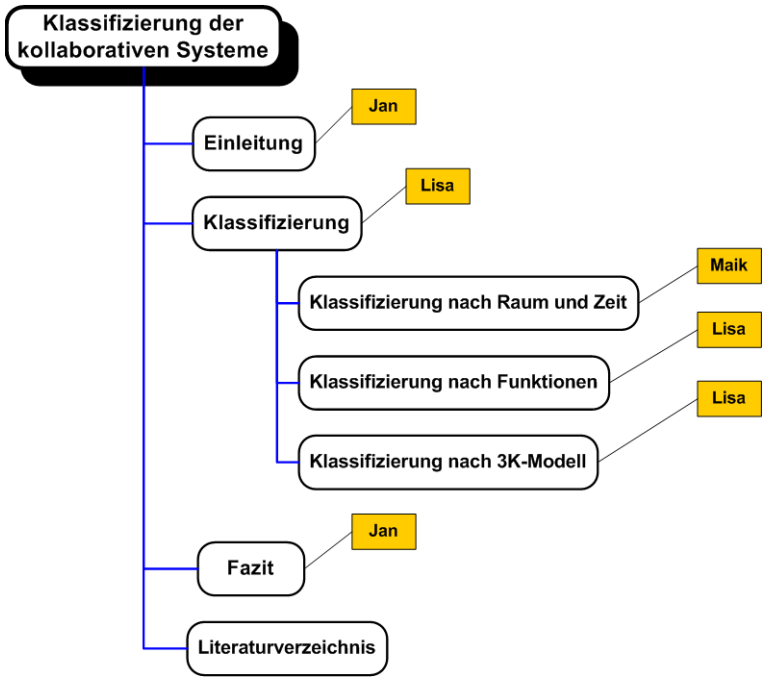

Abbildung 3.2: Dokumentenstruktur nach der ersten Phase des Szenarios

Nach dieser Start-Kollaboration gehen die Gruppenmitglieder zu der eigentlichen Dokumentenbearbeitung über, die bis zu mehreren Tagen in Anspruch nehmen kann. Dabei kann das Erstellen des Dokumentes für jeden beteiligten Teilnehmer in zwei wesentliche Phasen unterteilt werden, die sich gegenseitig mehrmals abwechseln können. Eine Phase besteht darin, den zugewiesenen

Abschnitt asynchron zu den anderen Gruppenmitgliedern zu editieren: den Text verfassen, weitere Unterabschnitte erstellen usw. Dabei können die neuen Abschnitte des Dokumentes wieder den anderen Teilnehmern zugewiesen werden. Außerdem können Notizen und Anmerkungen an die eigenen Abschnitte und an Abschnitte der anderen Gruppenmitglieder angefügt werden.

Die andere Phase beinhaltet die Kollaboration von zwei oder mehreren Gruppenmitgliedern. Diese Phase umfasst den Abgleich der Dokument-Versionen dieser Mitglieder untereinander. Solcher Abgleich kann zu Stande kommen, wenn zwei oder mehrere Teilnehmer miteinander in Verbindung treten, die entweder durch das World Wide Web, in einem lokalen Netzwerk oder ad hoc initialisiert werden kann.

Die Endphase der gemeinsamen Arbeit liegt in der Fertigstellung des Dokumentes, die das Beenden der Bearbeitung der einzelnen Gruppenmitglieder an ihren Teilaufgaben, das letzte Zusammenführen der Dokument-Versionen und das Abschließen des Dokumentes durch den Initiator beinhaltet. Die explizite Qualitätssicherung nach dem Beenden der Arbeit ist nicht notwendig, da die Gruppenmitglieder während des gesamten kollaborativen Prozesses die Möglichkeit hatten, die Qualität ihrer Texte gegenseitig zu kontrollieren.

Basierend auf dem Abschnitt 2.6 ergibt sich aus diesem Szenario, dass es sich um eine eher starke Verteilung der Gruppenmitglieder, eine dynamische Umgebung, einen sowohl online- als auch offline-Betrieb und eine eher spontane Kollaboration handelt.

3.2.2 Analyse der Aufgabe

Anhand des oben beschriebenen Szenarios können für das zu entwickelnde System folgende Anwendungsfall-Gruppen extrahiert werden:

- Verwaltung von kollaborativen Gruppen;
- Bearbeiten von Dokumenten;
- Rechte-Management.

In den folgenden Abschnitten werden diese Gruppen erläutert und die wichtigsten Anwendungsfälle vorgestellt, die zu diesen Gruppen gehören.

3.2.2.1 Verwaltung von kollaborativen Gruppen

Da die kollaborative Arbeit in einer Gruppe stattfinden soll, gehört die Gruppenverwaltung zu den Kernaufgaben des Systems. Aus dem Beispielszenario können folgende wesentliche, damit verbundene Anwendungsfälle abgeleitet werden:

- **Bildung einer Gruppe**
  Die Gruppe wird durch eine ausgezeichnete Person, den Initiator der Gruppe, erstellt. Dazu zählen die Vergabe eines Namen für die Gruppe und das Eintragen der zum Mitwirken an den Dokumenten dieser Gruppe berechtigten Teilnehmer.
- **Betreten der Gruppe**
  Die Bildung einer Gruppe setzt nicht voraus, dass andere Mitglieder diese Gruppe sofort betreten, so wie auch nicht vorausgesetzt werden kann, dass alle Gruppenmitglieder immer online sind. Dieser Vorgang kann von dem Interessenten zu einem beliebigen Zeitpunkt ausgelöst werden. Das Betreten der Gruppe beinhaltet die Authentifizierung des Interessenten, z.B. durch den Namen und das Passwort, und das Laden aller Dokumente dieser Gruppe, die zur Verfügung stehen.
- **Verlassen der Gruppe**
  Wenn ein Teilnehmer nicht mehr der Gruppe zugehören will oder kann, kann er die Gruppe verlassen. Dazu muss er seinen Wunsch dem Initiator der Gruppe mitteilen. Nur der Initiator kann die Mitgliedschaft dieses Teilnehmers auflösen, da er die Gruppe auch zusammengesetzt hat. Dementsprechend muss der Teilnehmer eine Verbindung zu dem Initiator aufbauen, wenn er aus der Gruppe austreten will. Durch den Initiator kann die Änderung der Gruppe dann an andere Team-Mitglieder propagiert werden.

3.2.2.2 Bearbeiten von Dokumenten

Das Hauptziel des zu entwickelnden Systems ist die Unterstützung der kollaborativen Bearbeitung von Dokumenten, dementsprechend zählt dieser Aufgabenbereich zu der Hauptfunktionalität des Systems. Der Akzent liegt dabei auf der Kollaboration, da dieser Aspekt der Dokumentenbearbeitung im Vergleich zu der Single-User-Dokumentenerstellung besondere Eigenschaften besitzt. Darüber hinaus soll das Entwickeln eines Editors für das eigentliche Schreiben von Texten nicht zu den Aufgaben des Systems gehören. Aus dem Beispielszenario können folgende Anwendungsfälle abstrahiert werden:

- **Erstellen eines neuen Dokumentes**
  Das Anlegen eines neuen Dokumentes geschieht in der Regel durch den Initiator der Gruppe, da die Gruppe zu dem Zweck erstellt wurde, die Dokumente zu bearbeiten. Das kann aber auch durch ein anderes Gruppenmitglied erfolgen, das danach als der Initiator des Dokumentes agiert. Abgesehen von der Vergabe eines Dokumententitels gehört zum Erstellungsprozess auch das Anlegen seines Grundgerüstes: der vorläufigen Abschnitte. Dieser Vorgang wird auch durch den Initiator des Dokumentes durchgeführt.

- **Verteilung der Aufgaben**
  Um das Dokument gemeinsam bearbeiten zu können, werden die einzelnen Abschnitte des Dokumentes jeweils einem Gruppenmitglied zugewiesen. Dadurch wird den Teilnehmern ein exklusives Recht auf die Bearbeitung eines oder mehrerer Dokumententeile vergeben. Durch diese Maßnahme wird der Entstehung von Inkonsistenzen im Dokument vorgebeugt (siehe Abschnitt 2.4). Bei dem Betreten der Gruppe durch ein Mitglied (siehe „Verwaltung von kollaborativen Gruppen“) werden die diesem Benutzer zugewiesenen Teile des Dokumentes durch das System identifiziert und für ihn entsprechend präsentiert, damit er sie bearbeiten kann.

- **Bearbeitung des Dokumentes**
  Zu der eigentlichen Bearbeitung des Dokumentes bzw. eines seiner Abschnitte gehört in erster Linie das Verfassen des entsprechenden Textes. Dazu soll das System dem Benutzer die Möglichkeit zur Verfügung stellen, den entsprechenden Teil des Dokumentes in einem beliebigen, von dem Benutzer ausgewählten Editor zu öffnen. Des Weiteren kann der Bearbeitende seinen Dokumentenbereich in weitere Abschnitte unterteilen und diese wiederum anderen Gruppenmitgliedern zuweisen (siehe vorheriger Anwendungsfall). In diesem Fall wird dieser Benutzer zum Initiator der neuen Dokumentabschnitte.

- **Annotationen an die Dokumentabschnitte**
  An die Dokumentabschnitte können im Laufe der Bearbeitung Notizen angebracht werden. Das kann entweder durch den Bearbeitenden selbst oder durch die anderen Gruppenmitglieder erfolgen. Die eigenen Notizen dienen dabei entweder zur Selbstkontrolle oder als Gedankenstützen für weitere Schritte. Durch die fremden Annotationen wird der kollaborative Prozess der Dokumentenbearbeitung unterstützt. Bei der Aktualisierung der lokalen Dokumenten-Kopie werden auch die Annotationen aktualisiert und durch das System entsprechend dargestellt.

- **Synchronisierung der Dokumente**
  Wenn mehrere Gruppenmitglieder online sind, können sie ihre lokalen Dokument-Versionen abgleichen. Der Synchronisierungsprozess kann entweder automatisch durch das System initiiert werden, wenn ein Mitglied die Gruppe betreten hat, oder durch ein Gruppenmitglied explizit aktiviert werden. Im Laufe dieses Prozesses werden alle neuen und aktuelleren Abschnitte und Annotationen von dem Kommunikationspartner abgerufen und die lokale Dokument-Kopie aktualisiert.

- **Beenden der Arbeit**
  Wenn ein Gruppenmitglied seine Arbeit an den ihm zugewiesenen Dokumententeilen abgeschlossen hat, kann er diese Teile als „fertig“ markieren. Bei einer Verbindungs-Gelegenheit werden diese Abschnitte an den Initiator des Dokumentes übertragen und das Editierrecht aufgehoben. Wenn alle Teile des Dokumentes abgeschlossen wurden und zu dem Initiator gelangen, kann auch das Dokument als „fertig“ markiert werden. Das kann durch den Initiator erfolgen.

3.2.2.3 Rechte-Management

Rechte-Management gehört zu den wichtigsten Funktionen des Systems. Darunter wird die Verwaltung von Zugriffsrechten der Benutzer auf die kollaborativen Dokumente verstanden. Aus dem Beispielszenario ergeben sich folgende Anwendungsfälle:

- **Vergabe der Editierrechte**
  Wie schon in vorherigen Abschnitt besprochen wurde, kann ein Gruppenmitglied einen Teil des Dokumentes bearbeiten, wenn er die entsprechende Berechtigung von dem Initiator dieses Teils erhalten hat. Das System soll dementsprechend die Kontrolle über die Rechtevergabe und die Zugriffe auf die Dokumente übernehmen.

- **Aufheben der Editierrechte**
  Wenn ein Gruppenmitglied relativ lange Zeit nicht erreichbar ist (hat sich lange nicht gemeldet), können seine Schreibrechte aufgehoben werden. Diese Maßnahme ist insofern nötig, da das Ausfallen eines Teilnehmers keine negative Auswirkung auf den Bearbeitungsfluss des Dokumentes haben soll. Wie schon in vorherigen Abschnitten besprochen wurde, hat die Erstellung eines Dokumentes im Normalfall einen bestimmten zeitlichen Rahmen, der nicht überschritten werden soll. Aus

diesem Grund muss der Initiator des Dokumentes oder eines Dokumententeils in der Lage sein, die Editierrechte des nicht erreichbaren Mitglieds aufzuheben und sie einem anderen Beteiligten zu vergeben. Dementsprechend muss das System die Präsenz der Gruppenmitglieder überwachen und im Stande sein, eventuelle Konfliktsituationen zu lösen, wenn sich der „verschwundene“ Teilnehmer doch später meldet.

In Bezug auf die Rolle des Initiators bei der Erstellung eines Dokumentes sind zwei Varianten des Szenarios möglich. Dementsprechend unterscheidet sich auch der Gesamtablauf der Dokumentenbearbeitung. Zum einen kann eine Person mehr Berechtigung aufweisen als andere Beteiligten. Das ist der Fall, wenn z.B. ein Professor mit seinen Studenten einen wissenschaftlichen Artikel verfasst. In diesem Fall gelten folgende Rahmenbedingungen:

- Diese Person ist der Initiator.
- Abgeschlossene Aufgaben müssen an sie übergeben werden, da diese Person für die Fertigstellung des Dokumentes verantwortlich ist und nur sie weitere eventuell notwendige Schritte, z.B. für die Veröffentlichung des Dokumentes, einleiten kann.
- Ausfall des Initiators wird dem Stoppen der Bearbeitung gleichgesetzt. In diesem Fall kann das Dokument nicht erfolgreich abgeschlossen werden, da die verantwortliche Einheit fehlt, für die das Dokument erstellt werden sollte.
- Übergabe des Initiator-Rechtes an eine andere Person muss gewährleistet werden. Das aber soll nur durch den aktuellen Initiator erfolgen können.

Bei der anderen Variante der Dokumentenerstellung sind alle beteiligten Personen gleichberechtigt, z.B. wenn ein Dokument durch mehrere Studenten verfasst wird, die den gleichen Grad der Verantwortung für das Endergebnis tragen. In diesem Fall gilt:

- Die Wahl des Initiators erfolgt auf freiwilliger Basis. Als Initiator kann theoretisch jeder agieren.
- Für den Fall des Initiator-Ausfalls wird die Möglichkeit gegeben, einen neuen zu bestimmen, damit die Bearbeitung des Dokumentes nicht abgebrochen wird.
- Übergabe des Initiator-Rechtes an eine andere Person durch den aktuellen Initiator muss gewährleistet werden.

3.2.3 Analyse des zu entwickelnden Systems

Aus den beschriebenen Anforderungen (dynamische Umgebung, spontane Kollaboration, Online- und Offline-Arbeit) geht hervor, dass die Beteiligten uneingeschränkt in ihren Handlungen und trotzdem kollaborativ ein gemeinsames Dokument bearbeiten wollen. Dementsprechend soll das unterstützende System sehr flexibel in Bezug auf die Arbeitsumgebung sein. Wie in der Vision beschrieben wurde, hat eine leichtgewichtige, dezentrale Architektur mehr Potenzial unter diesen Bedingungen, als eine zentralisierte System-Lösung (siehe Abschnitt 2.3.4). Aus diesem Grund wird für das zu entwickelnde Editiersystem eine Peer-to-Peer-Architektur gewählt.

Das System wird die Verwaltung der Gruppenmitglieder und der Dokumente übernehmen wie auch die Aufgabenverteilung unterstützen. In diesem System wird jeder Teilnehmer der Gruppe eine Kopie des ganzen gemeinsamen Dokumentes besitzen, damit jeder seine persönlichen Teilaufgaben asynchron bearbeiten kann. Dieses Vorgehen garantiert die Ausfallsicherheit des Systems und die Möglichkeit, auch offline das Dokument zu bearbeiten. Außerdem erhöht diese Strategie die kollaborative Awareness, da die Beteiligten dadurch die Kenntnis über die gemeinsamen Bearbeitungsfortschritte besitzen. Des Weiteren soll das Editiersystem die Synchronisation der Dokumenten-Kopien unterstützen und die Kontrolle über die Zugriffsrechte übernehmen.

Aus dem oben Beschriebenen kann extrahiert werden, dass das zu entwickelnde kollaborative System folgende Hauptfunktionalität unterstützen soll:

- Verwaltung von Gruppen und Gruppenmitgliedern;
- Dokumenten-Erstellung;
- Synchronisierung der Dokumente;
- Rechte-Verwaltung;
- Unterstützung der Kommunikation innerhalb der Gruppe.

Abgesehen von dem Erfüllen der funktionalen Anforderungen soll im Rahmen des System-Designs ein strukturiertes Format für Dokumente entwickelt werden, das eine feine Granularität der Aufgabenverteilung ermöglichen wird.

Da das Ziel dieser Arbeit nicht in der Entwicklung einer Peer-to-Peer-Plattform liegt, wird angestrebt, für die Kommunikation eine existierende Plattform in das Gesamtkonzept zu integrieren. Wie schon im vorherigen Kapitel beschrieben wurde (Abschnitt 2.3.4), ist JXTA zur Zeit das am meisten

ausgereifte Peer-to-Peer-Framework. Aus diesem Grund wird für die Unterstützung der Kommunikation diese Plattform eingesetzt.

Kapitel 4

Design und Realisierung

4.1 Entwurf der Dokumentenstruktur

Wie im Kapitel „Grundlagen“ (Abschnitt 2.1) erläutert wurde, ist ein strukturiertes Format besonders geeignet für kollaborative Dokumente. Ein kollaboratives Dokument sollte als eine komplexe Struktur betrachtet werden, die aus mehreren logischen Teilen besteht (siehe Abbildung 4.1). Diese Teile unterscheiden sich voneinander durch den Typ der darin enthaltenen Informationen und Daten. Zum einen gehört dazu der eigentliche fachliche Inhalt des Dokumentes, z.B. der Inhalt einer wissenschaftlichen Arbeit. Zum anderen besitzt das Dokument Informationen über seine Struktur (Kapitel, Unterkapitel usw.) und Formatierung, die so genannten Meta-Informationen oder Meta-Daten. Letztendlich muss ein solches Dokument auch kollaborative Informationen enthalten. Zu diesen Informationen gehören solche Daten wie Autor des Dokumentes und der einzelnen Abschnitte, Datum der Erstellung, Editierrechte etc.

Es gibt mehrere Möglichkeiten, die beschriebenen Teile des Dokumentes auf eine Dokumentenstruktur abzubilden. Zum einen können alle Informationen und Daten in einer einzelnen Struktur repräsentiert werden. Solche Struktur wird dann sowohl den fachlichen Text des Dokumentes als auch seine Meta- und kollaborative Daten enthalten. Die andere Variante besteht darin, diese drei logischen Teile des kollaborativen Dokumentes voneinander zu trennen. Dieses Vorgehen ist insofern vorteilhafter als die erste beschriebene Variante, da dadurch eine höhere Flexibilität in Bezug auf Formatierung und Darstellungsmöglichkeiten des Dokumentes erreicht werden kann. So kann ein Dokument mehrere Formatierungsvorlagen besitzen, die nach Bedarf verwendet werden könnten, ohne dabei viele Änderungen an dem Dokument

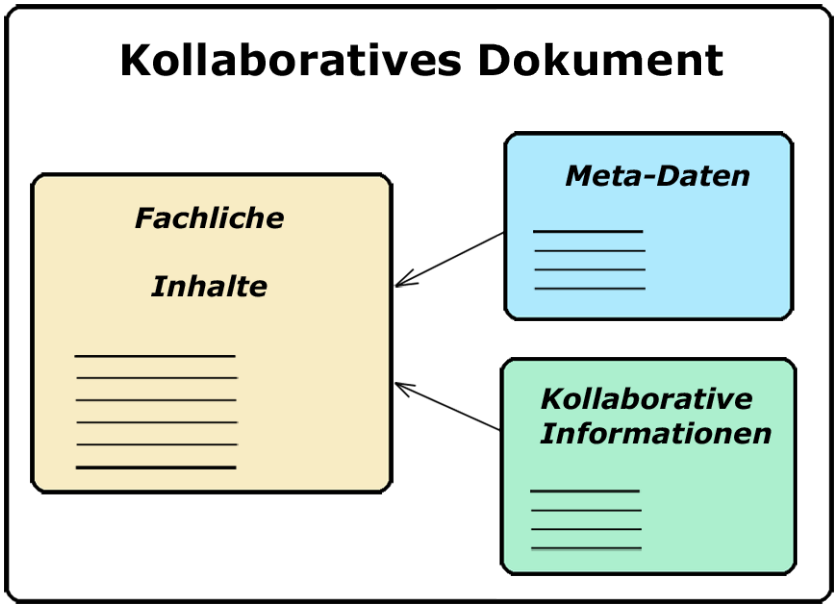

Abbildung 4.1: Aufbau des kollaborativen Dokumentes

vornehmen zu müssen. Darüber hinaus wird es möglich sein, für die fachlichen Inhalte des Dokumentes ein schon existierendes, bekanntes Format zu verwenden. Für die Bearbeitung der Texte könnte dementsprechend ein für dieses Format geeigneter Editor benutzt werden. Abgesehen davon wäre das Editiersystem flexibler, da es im Stande wäre, verschiedene Dokumentenformate zu unterstützen.

Von anderer Seite sollen die kollaborativen Informationen des Dokumentes vor dem Benutzer verborgen bleiben, d.h. er soll keine Möglichkeit haben, diese Daten zu ändern oder zu löschen. Bei der Trennung des Kollaborativen von dem Fachlichen kann das leichter erreicht werden, da in diesem Fall kein Zwischenschritt nötig ist, der den fachlichen Inhalt eines Dokumententeils von den anderen Daten der Dokumentenstruktur abstrahiert, um dem Benutzer den entsprechenden Text für die Bearbeitung zur Verfügung zu stellen.

Basierend auf diesen Überlegungen wird entschieden, die zweite beschriebene Variante der Dokumentenstruktur für das kollaborative Editiersystem zu entwickeln und für den fachlichen Inhalt der Dokumente ein schon existierendes Dokumentenformat einzusetzen.

4.1.1 Logische Aufteilung des Dokuments

Der Grad und die Art der Trennung zwischen den einzelnen Teilen der Dokumentenstruktur können variieren. In der Abbildung 4.2 ist die Dokumentenstruktur schematisch dargestellt worden, die in dieser Arbeit für kollaborative Dokumente entworfen wurde.

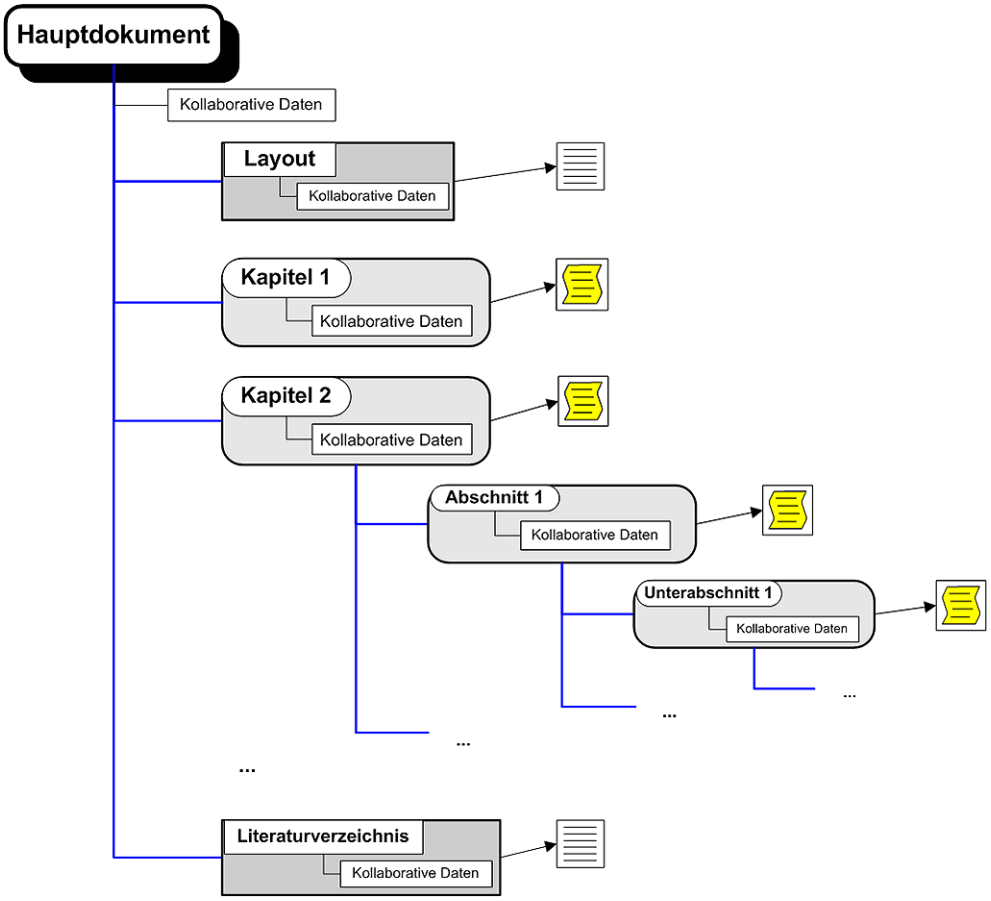

Abbildung 4.2: Struktur des kollaborativen Dokumentes

Nach diesem Entwurf besteht das kollaborative Dokument aus drei wesentlichen Teilen: dem Hauptdokument, den fachlichen Inhalten und seinen Formatierungsvorlagen. Das *Hauptdokument* (siehe Abbildung) bildet das Grundgerüst des Dokumentes. Es hat einen hierarchischen Aufbau und spiegelt die logische Struktur des Dokumentes wider: seine Kapitel, Unterkapitel und weitere Abschnitte sowie das Literaturverzeichnis, das Glossar usw. Die eigentlichen fachlichen Inhalte der einzelnen Abschnitte werden separat gespeichert,

das Hauptdokument beinhaltet nur Verweise darauf. In Abbildung 4.2 sind die fachlichen Inhalte als Rechtecke mit einem Blatt Papier schematisch dargestellt. Die Verweise aus dem Hauptdokument darauf sind durch die Pfeile gekennzeichnet.

Den weiteren Teil des kollaborativen Dokumentes bilden seine eventuellen Formatierungsvorlagen, die in der Abbildung 4.2 als *Layout* dargestellt sind. Ähnlich den fachlichen Inhalten werden die Layout-Informationen getrennt von der Hauptstruktur gespeichert, das Hauptdokument hält nur eine Referenz darauf.

Letztendlich enthält das Hauptdokument die kollaborativen Informationen des Dokumentes. Diese Informationen haben eine feine Granularität, d.h. dass abgesehen von den kollaborativen Daten des Gesamtdokumentes auch jeder seiner Abschnitte diese Informationen enthält. Sie werden in entsprechenden Knoten der Hauptdokument-Struktur festgehalten (siehe Kästchen *Kollaborative Daten* im Bild).

4.1.2 Dokumentenformat

Ein weiterer wichtiger Punkt im Entwurf der kollaborativen Struktur ist die Entscheidung, welches Format die Dokumente haben werden. Wie in den vorherigen Abschnitten beschrieben wurde, besteht die Dokumentenstruktur aus drei Teilen. Dabei sind von besonderer Bedeutung das Hauptdokument und die fachlichen Inhalte. Da diese Komponenten des Dokumentes getrennt voneinander gehalten werden, können verschiedene Formate für sie verwendet werden. In diesem Abschnitt werden geeignete Formate analysiert und die Entscheidung getroffen, welche von ihnen für das kollaborative Dokument eingesetzt werden. Das Layout-Format steht in einer direkten Abhängigkeit von dem Format der fachlichen Inhalte, deswegen werden relevante Formate für die Formatierungsvorlagen nicht separat behandelt.

Für die fachlichen Inhalte können theoretisch beliebige Formate verwendet werden. In dieser Arbeit wird aber angestrebt, kein proprietäres Dokumentenformat zu nehmen, sondern das Editiersystem für offene, textuelle und strukturierte Standards zu konzipieren, die vor allem fähig sind, ein Dokument aus mehreren physikalischen Teilen (Dateien) zusammenzusetzen. Dazu gehören in erster Linie LaTeX und DocBook (siehe Abschnitt 2.1). Hier soll ein Vergleich dieser beiden Dokumentenformate durchgeführt werden. Aufgrund der Ergebnisse dieses Vergleichs wird entschieden, welches Format im zu entwickelnden System eingesetzt wird.

DocBook ist ein sehr umfangreiches, XML-basiertes Format. Demzufolge

müssen der Text des Dokumentes sowie alle besonderen Textauszeichnungen, wie Absätze, Fett-Druck, Kapitel oder Aufzählungen, im DocBook als XML-Tags repräsentiert werden. Darin liegen seine Stärken wie auch Schwächen. Zu den Stärken kann die strikte Strukturierung gezählt werden, die eine gute Übersichtlichkeit der Dokumentenstruktur bietet. Die Nachteile bilden dagegen der damit verbundene Schreibaufwand und eher schwache Lesbarkeit der Dokument-Inhalte. LaTeX ist im Vergleich zu DocBook ein einfacheres textuelles Format, in dem besonders zu formatierende Stellen und die Dokumentenstruktur mit Hilfe eines Satzes von textuellen Befehlen (Makros) ausgezeichnet werden können. Dokumente in diesen beiden Formaten lassen sich mit herkömmlichen Texteditoren bearbeiten, obwohl sich das Schreiben von DocBook-Dokumenten in einem einfachen Editor wegen seiner XML-Struktur aufwändiger gestaltet als das Bearbeiten eines LaTeX-Dokumentes. Darüber hinaus existieren sowohl für LaTeX als auch für DocBook mehrere Prozessoren, die aus diesen Eingabe-Formaten gängige Ausgabe-Formate produzieren können, wie z.B. PDF, HTML oder PostScript. Bei DocBook wird das durch die Verwendung eines entsprechenden XSL[1]-Stylesheets, eines XSLT[2]-Prozessors und, wenn nötig, einer entsprechenden XSL-FO[3]-Engine möglich [Trieloff]. In LaTeX wird die benötigte Ausgabe durch den Einsatz des entsprechenden Konvertierungs-Werkzeuges zur Erzeugung von PostScript-, PDF- oder HTML-Dateien erreicht.

Die beschriebenen Eigenschaften der beiden Dokumentenformate sind besonders aus der Sicht des Anwenders wichtig. Für das kollaborative System stehen aber andere Kriterien im Vordergrund. Dazu zählt in erster Linie die Fähigkeit, das zu bearbeitende Dokument aus mehreren separaten Dateien, die die einzelnen Abschnitte des Dokumentes repräsentieren, zusammenzusetzen. Beide Formate, sowohl DocBook als auch LaTeX, ermöglichen dies. Bei DocBook stehen für die Aufteilung des Dokumentes auf mehrere Dateien (Modularisierung) zwei Möglichkeiten zur Verfügung. Entweder kann das über die externen Entitäten oder mit Hilfe des Standards *XInclude* durchgeführt werden [Trieloff]. Der Einsatz von externen Entitäten ist mit einigen Nachteilen verbunden, zu denen vor allem die notwendige Anpassung der DTD-Einbindung gehört. Dadurch sind keine dynamischen Modularisierungen der Dokumententeile möglich. Mit der Verwendung des *XInclude*-Standards entstehen diese Probleme nicht. Der Nachteil liegt hier aber darin, dass nicht alle Verarbeitungsprogramme die Auflösung von *XInclude* unterstützen [Trieloff]. In LaTeX ist die Einbindung der externen Dokumen-

[1]Extensible Style Language
[2]XSL Transformations
[3]XSL-Formatting Objects

tenteile sehr einfach. Dazu existieren zwei Makros: *input* und *include*. Der Unterschied zwischen diesen zwei Befehlen liegt darin, dass *include*, abgesehen von dem unterschiedlichen Formatierungs-Verhalten, keine weitere *include*-Verschachtelungen in den eingebundenen Teildokumenten erlaubt. Das rekursive Einbinden von weiteren Teilen mit *input* ist dagegen in beiden Fällen möglich.

Nach diesem Vergleich wird deutlich, dass LaTeX im Vergleich zu DocBook einfacher in der Anwendung ist. Obwohl diese beiden Formate sich in vielerlei Hinsicht voneinander unterscheiden, haben sie eine Gemeinsamkeit: Sie sind nicht für die Unterbringung von kollaborativen Informationen konzipiert. Abgesehen aber von diesem Umstand ist die Trennung zwischen den sachlichen Inhalten und den kollaborativen Daten auf jeden Fall vorzuziehen (siehe vorige Abschnitte). Aufgrund dieser Überlegungen und der Tatsache, dass die Verwendung von LaTeX im Gegensatz zu DocBook sehr verbreitet in Hochschulkreisen ist, wird für fachliche Dokumenteninhalte das LaTeX-Format eingesetzt.

Aus Gründen, die ausführlich im Abschnitt 2.1 beschrieben wurden, wird für das Hauptdokument ein XML-basiertes Format verwendet.

4.1.3 Zugriffsrechte innerhalb des Dokumentes

Eine besondere Stelle wird hier dem Thema der Zugriffsrechte zugeteilt, da es zu den wichtigsten Aspekten eines kollaborativen Systems gehört. In dem zu entwickelnden System werden kollaborative Dokumente eine zentrale Rolle spielen, deswegen wird die ausgewählte Zugriffsrechtestrategie auch eine Auswirkung auf die Dokumentenstruktur haben. In Bezug auf die Dokumentenstruktur sind folgende Fragen von Bedeutung: Welche Zugriffsrechte machen in diesem System Sinn und wo sollen sie verwaltet werden? In diesem Abschnitt werden diese beiden Fragen geklärt.

In dieser Arbeit wird von einer gutwilligen Umgebung ausgegangen, in der die einzelnen Gruppenmitglieder an den Dokumenten arbeiten, ohne Absichten, den anderen zu schaden. Unter diesem Kontext werden als Bearbeitungsobjekte die einzelnen Dokumententeile verstanden und als Zugriffsrechte die Rechte, einen Dokumententeil zu erstellen, zu editieren und zu löschen. Da die Gruppenmitglieder gemeinsam an einem Dokument arbeiten, um eine gemeinsame Aufgabe durchzuführen, wird davon ausgegangen, dass jeder Teilnehmer die Dokumentabschnitte der anderen Gruppenmitglieder ansehen darf. Aus diesem Grund wird die Verteilung von Lese-Zugriffsrechten in einer Gruppe nicht benötigt. Die Schreibrechte werden dagegen als ein Teil

der kollaborativen Informationen angesehen und dienen dazu, dem Benutzer zu vermitteln, welche Teile des Dokumentes ihm zur Bearbeitung freigegeben sind und welche nicht, damit keine Inhaltsinkonsistenzen im Dokument entstehen.

Abgesehen von dem eigentlichen Schreiben an einem Dokumententeil hat ein Gruppenmitglied das Recht, neue Dokumententeile zu erstellen und die Schreibrechte daran anderen Gruppenmitgliedern zuzuweisen (siehe Abschnitt 3.2 „Aufgabenstellung“). Dementsprechend kann es bei einem Dokumententeil dazu kommen, dass der Besitzer dieses Rechtes nicht mit dem Besitzer des Schreibrechtes übereinstimmt. Aus diesem Grund wird für Dokumente und Dokumententeile ein zusätzliches Recht benötigt, das so genannte Eigentümer-Recht (Die Rechtevergabe-Strategie wird im Abschnitt 4.2 beschrieben).

Da das zu entwickelnde System eine Peer-to-Peer-Architektur ohne eine zentrale Komponente haben wird, werden die Editierberechtigungen als Teil der Dokumentenstruktur angesehen. Das bedeutet, dass jeder Teil des kollaborativen Dokumentes die Information darüber besitzen wird, welcher Benutzer der Eigentümer dieses Teils ist und welcher das Schreibrecht daran hat.

4.1.4 Kollaborative Informationen

Aus den oben beschriebenen Überlegungen und durch die im Abschnitt 3.2.2 beschriebenen Design-Anforderungen kann festgelegt werden, welche kollaborativen Informationen das Dokument enthalten soll. Dazu gehören folgende Daten:

- *Eindeutige Identifizierung des Dokumentes und seiner Abschnitte*
- *Autor des Dokumentes oder des Abschnitts*: Diese Eigenschaft soll die Person identifizieren, die den Teil des Dokumentes erstellt hat und die Berechtigung hat, weitere Unterabschnitte zu erstellen und die Editierrechte an diesen Abschnitten an andere Personen weiterzugeben.
- *Aktuelle Version*: Diese Information ist für die Synchronisierung der Dokumente notwendig. Damit wird festgestellt, ob die zu vergleichenden Abschnitte synchronisiert werden müssen oder nicht.
- *Editierrechte*: Nur die Person, die das Editierrecht an einem Dokumentabschnitt besitzt, darf den Inhalt dieses Abschnitts ändern.

- *Annotationen*: Diese Eigenschaft ermöglicht, zusätzliche Ergänzungen oder Korrekturen an die Dokumentabschnitte anzubringen, zu denen man unter Umständen keine Editierberechtigung besitzt.
- *Editierstatus* (in Bearbeitung oder abgeschlossen): Das Attribut ist wichtig bei der Neu-Vergabe des Editierrechtes. Nur für abgeschlossene Abschnitte kann dieses Recht geändert werden.
- *Existenz-Status* (gelöscht oder nicht): Diese Eigenschaft ist für die Synchronisierung des Dokumentes nötig. Dadurch wird signalisiert, dass der auf einer Seite nicht existierende Abschnitt gelöscht wurde. Ohne diese Eigenschaft könnte nicht festgestellt werden, ob der Abschnitt von einem Benutzer gelöscht wurde oder von einem anderen Benutzer erstellt wurde.
- *Datum der Erstellung* ist für die Historie des Dokumentes nützlich.
- *Datum der letzten Änderung* wird auch für diesen Zweck benutzt.

4.1.5 XML-Schema des kollaborativen Dokumentes

Anhand der beschriebenen Überlegungen wurde ein Schema für das kollaborative Dokument entworfen, das in der Abbildung 4.3 zu sehen ist.

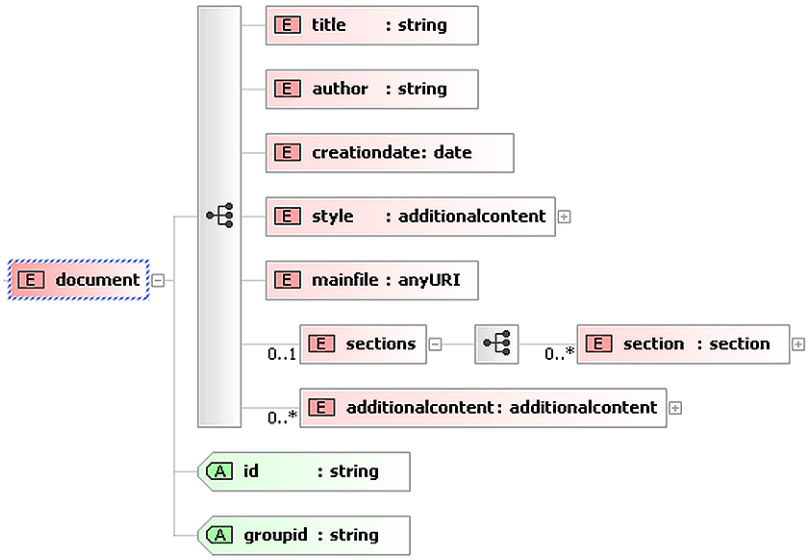

Abbildung 4.3: XML-Schema für kollaborative Dokumente

Document stellt nach diesem Entwurf das Haupt-Element des Schemas (Root-Element) dar. Abgesehen von den kollaborativen Informationen (*id*, *author* und *creationdate*) und dem Titel besitzt das *Document*-Element unter anderem das Attribut *groupid* für eine eindeutige Zuordnung des Dokumentes zu einer kollaborativen Gruppe. Darüber hinaus enthält das Document das Element *mainfile*. Dieses Element soll die Referenz auf die Meta-Daten der fachlichen Inhalte des Dokumentes beinhalten. Diese Daten sollen beschreiben, aus welchen Teilen das fachliche Dokument zusammengesetzt ist, und sind für die Bearbeitung des Dokumentes außerhalb des kollaborativen Systems nötig.

Des Weiteren enthält das Dokument das *section*-Element (siehe Abbildung). *Section* dient zum eigentlichen logischen Aufbau des Dokumentes. Dieses Element stellt Kapitel, Unterkapitel und weitere Teile des Dokumentes dar. Da der logische Aufbau des Dokumentes eine Verschachtelung der Abschnitte aufweisen kann, muss das *section*-Element eine rekursive Struktur besitzen. Dementsprechend kann es weitere *section*-Elemente beinhalten, die ihrerseits

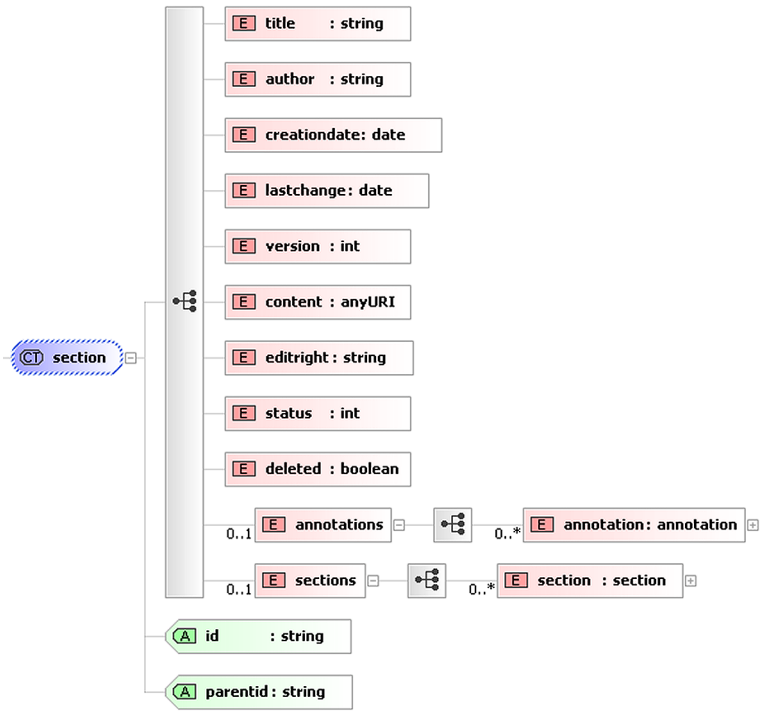

Abbildung 4.4: Schema für das *section*-Element

auch Unterelemente haben können, usw. Des Weiteren können an einen Dokumentabschnitt Notizen angebracht werden. Im Design des *section*-Elementes

soll das ebenfalls entsprechend berücksichtigt werden. Auf der Grundlage dieser Überlegungen wurde ein neuer XML-Typ für die *section*-Elemente definiert, der in der Abbildung 4.4 dargestellt ist. Dieser Typ besitzt demnach Elemente für kollaborative Informationen, für den Titel des Abschnitts, für weitere Unterabschnitte und für die Annotationen zu diesem Abschnitt. Außerdem hat dieses Element zwei Attribute: *id* für eine eindeutige Identifizierung eines Abschnitts und *parentid* für die Zuordnung eines Abschnitts zu seinem „Vater"-Abschnitt. Diese Eigenschaft wird für die Synchronisierung von neuen Teilen des Dokumentes benötigt (mehr zur Synchronisierung im Abschnitt 4.4.3).

Zu den weiteren Teilen des Dokumentes gehören Elemente *style* und *additionalcontent*. Das *style*-Element beinhaltet Daten der Formatierungsvorlage des Dokumentes. Unter dem *additionalcontent* werden so genannte zusätzliche Inhalte des Dokumentes verstanden: Literaturverzeichnis, Glossar etc. Die Eigenschaften, die für die Beschreibung dieser beiden Teile des Dokumentes benötigt werden, unterscheiden sich von den Daten, die für die Darstellung der anderen Abschnitte des Dokumentes notwendig sind. So z.B. werden an eine Formatierungsvorlage keine Notizen angebracht. Aus diesem Grund wurde für diese beiden Elemente ein neuer XML-Typ definiert, der in der Abbildung 4.5 zu sehen ist.

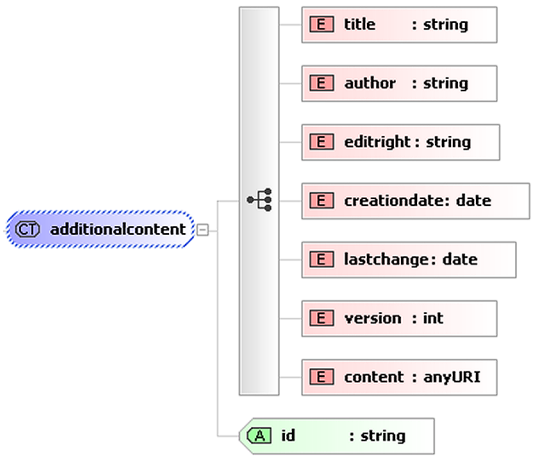

Abbildung 4.5: Schema für das *additionalcontent*-Element

Das *annotation*-Element beschreibt Notizen (Anmerkungen, Ergänzungen, Korrekturen usw.) zu den Dokumentabschnitten. Abgesehen von den kollaborativen Daten enthält dieses Element den Titel und den Text der jeweiligen Notiz und eine Eigenschaft, die eine Aussage darüber macht, ob die

Anmerkung abgearbeitet (oder in Kenntnis genommen) wurde. Für dieses Element wurde auch ein neuer Typ definiert (siehe Abbildung 4.6). Der *annotation*-Typ enthält keine Id. Da alle an der Bearbeitung des Dokumentes beteiligten Gruppenmitglieder ihre Anmerkungen an alle Dokumententeile anbringen können, wird eine Id als eindeutiges Attribut nicht ausreichen. Aus diesem Grund wird die Kombination aus dem Titel der Notiz und ihrem Autor als eine eindeutige Identifizierung genommen.

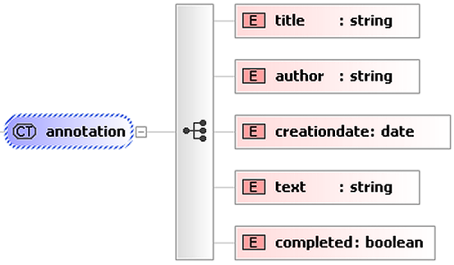

Abbildung 4.6: Schema für das *annotation*-Element

4.2 Editierrechte-Strategie

Das Thema des Zugriffsrechte-Managements ist sehr umfangreich und komplex. Dieser Bereich beschäftigt sich mit den Konzepten und Lösungen für die effiziente und sichere Vergabe und Verwaltung von Berechtigungen auf die Bearbeitungsobjekte. Es gibt mehrere Aspekte, die bei der Entwicklung eines Berechtigungsmodells berücksichtigt werden müssen, so z.B. Feingranularität von Zugriffsrechten, Vererbung von Berechtigungen, Kumulation der Rechte usw. (mehr dazu in [Mund]). Es existieren drei gängige Zugriffskontrollstrategien-Klassen: Rollen-basierte Zugriffskontrollstrategien (engl. *Role Based Access Control*), System-bestimmte Strategien (engl. *Mandatory Access Control*) und Benutzer-bestimmbare Strategien (engl. *Discretionary Access Control*) (nach [Eckert]). Sie unterscheiden sich in den angewendeten Ansätzen und dem Grad der Kontrolle, die durch das System vorgesehen wird. Für diese Strategien existieren ihrerseits mehrere Zugriffskontrolle-Modelle, z.B. Zugriffsmatrix-Modelle, rollenbasierte Modelle usw. (nach [Eckert]).

Die Entwicklung eines umfassenden Berechtigungsmodells würde den Rahmen dieser Arbeit überschreiten. Aus diesem Grund wird nur eine Strategie

für die Vergabe der Editierrechte innerhalb eines Dokumentes als ein Teil des gesamten Zugriffskontrolle-Modells ausgearbeitet.

Ein verbreiteter Ansatz für die Berechtigung-Vergabe ist die Vergabe der Editierrechte erst nach ihrer Anforderung durch einen Benutzer mit dem Bearbeitungswunsch. Im Zusammenhang mit dem dezentralen und dynamischen Charakter des Systems erscheint dieser Ansatz aber ineffizient. Da keine zentrale Stelle im System existiert, die das Verwalten solcher Anfragen übernehmen würde, kann es dazu kommen, dass das Gruppenmitglied mit dem Bearbeitungswunsch der Einzige ist, der sich zur Zeit online befindet. Das kann dazu führen, dass dieses Mitglied keine Möglichkeit bekommt, das Dokument zu editieren, da niemand ihm die Zugriffsberechtigung erteilen kann. Außerdem würde ein Gruppenmitglied nur dann die Möglichkeit zum Editieren haben, wenn es online ist, da es jedes Mal die Berechtigung dafür anfordern muss.

Ein anderer Ansatz besteht darin, das Editieren ohne eine explizite Erteilung der Berechtigung zu erlauben. Im schlimmsten Fall können durchgeführte Änderungen mit den Änderungen anderer Gruppenmitglieder kollidieren, wenn sie am gleichen Textabschnitt gemacht wurden. Dementsprechend ist entweder ein Merging-Konzept nötig oder die Änderungen müssen später per Hand zusammengeführt werden. Von anderen Seite kann das Bearbeiten eines Dokumentabschnitts durch mehrere Teilnehmer dazu führen, dass im Textfluss Unstimmigkeiten auftreten. Darüber hinaus könnte es zu Unzufriedenheit der einzelnen Bearbeitenden kommen.

Wie schon im Abschnitt 3.1 beschrieben wurde, beinhaltet die Bearbeitung eines Dokumentes durch mehrere Personen sehr oft die Aufteilung der Aufgaben unter den Beteiligten. Diese Vorstellung wurde als Grundlage für die Editierrechte-Strategie im dem zu entwickelnden System genommen.

Bei einer klaren Verteilung der Aufgaben können für die einzelnen Dokumententeile langfristige Editierrechte zugewiesen werden, die als eine Art der dauerhaften Sperren angesehen werden können. Die Rechte werden durch die Gruppenmitglieder gegenseitig vergeben. Ein Gruppenmitglied bekommt die Editierrechte schon im Vorfeld der Bearbeitung eines Dokumententeils und behält diese Rechte für ein langes Zeitintervall, bis es die Bearbeitung abgeschlossen hat und diese Rechte dadurch selbst freigibt. Dadurch existiert für einen Dokumentabschnitt zu einem Zeitpunkt nur ein Gruppenmitglied, das die Editierrechte auf diesen Abschnitt besitzt. Es können also keine Inkonsistenzen im Dokument entstehen. Dementsprechend gibt es keinen Bedarf in der Ausarbeitung eines umfangreichen Merging-Verfahrens. Außerdem kann das Gruppenmitglied bei dieser Strategie auch offline seinen Teil des Doku-

mentes bearbeiten und ist auf andere Gruppenmitglieder nicht angewiesen. Die im vorherigen Abschnitt ausgearbeitete Dokumentenstruktur erlaubt eine fein-granulare Aufteilung der Editierrechte und das Anbringen von Annotationen an die Dokumentabschnitte, wodurch die Gruppenmitglieder nicht voneinander isoliert werden und der Gesamt-Kollaborationsprozess nicht gestört wird.

Es wird folgende Strategie angewendet. Die Editierrechte auf einen Dokumententeil bleiben vorerst der Person vorbehalten, die diesen Dokumententeil angelegt hat (dem Autor). Diese Person ist berechtigt, die Editierrechte an eine weitere Person zu vergeben. Bei der Übertragung der Editierrechte auf einen Abschnitt werden rekursiv Rechte auf alle seinen Unterabschnitte entsprechend geändert. Davon ausgeschlossen sind Unterabschnitte, die schon an eine dritte Person übertragen wurden. Die andere Person hat ihrerseits die Möglichkeit, neue Unterabschnitte in dem ihr zugewiesenen Dokumententeil zu erstellen und die Editierrechte auf diese Abschnitte wieder weiterzuvergeben usw. Folglich wird im System eine Strategie der rekursiven Rechtevergabe verwendet. Bei der Aufhebung der Editierberechtigung (siehe 3.2.2) werden die Editierrechte auf den Autor des entsprechenden Abschnitts zurück übertragen. Die Editierrechte darf demzufolge nur der Autor der entsprechenden Dokumententeile aufheben.

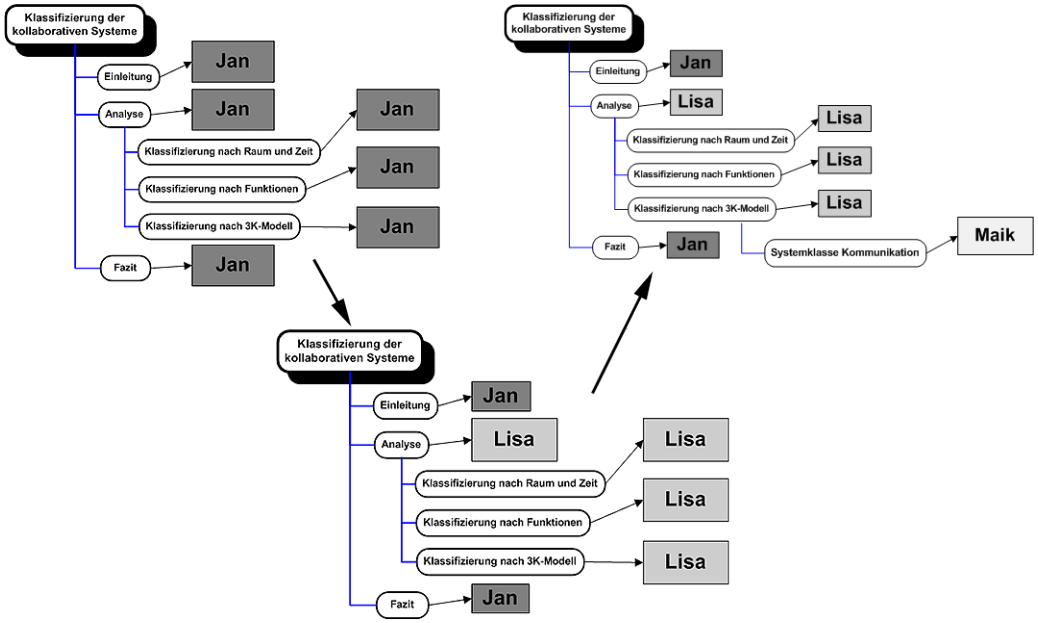

Abbildung 4.7: Rechtevergabe an einem Beispiel

In der Abbildung 4.7 ist diese Strategie an einem Beispiel vorgestellt. Die Abbildung präsentiert den Dokumentbaum aus dem Szenario (Abschnitt 3.2.1) in drei Editierrechte-Zuständen. Die auf einen Dokumententeil zugewiesenen Editierrechte sind durch die rechteckigen Kästchen mit dem Namen der entsprechenden Person dargestellt. Das erste Bild zeichnet den Zustand nach der Erstellung des Dokumentes: Editierrechte an allen Dokumententeilen besitzt *Jan*, der das Dokument und seine Struktur angelegt hat. Das zweite Bild stellt den Zustand dar, nachdem Jan die Rechte an dem Abschnitt „Analyse“ *Lisa* zugewiesen hat. Dementsprechend besitzt *Lisa* die Editierrechte auch auf die Unterabschnitte der „Analyse“. Das letzte Bild zeigt die Rechte-Verteilung, nachdem Lisa ein neues Abschnitt erstellt hat und die Editierrechte darauf an *Maik* übertragen hat.

4.3 Überblick über die Architektur

Aus dem im Kapitel „Analyse“ beschriebenen Beispielszenario und der darauf folgenden Analyse der Aufgabe wird ersichtlich, dass das zu entwerfende kollaborative Editiersystem eine Anwendung umfasst, die bei jedem beteiligten Teilnehmer laufen soll. Das System besteht aus vier großen Teilen, die in der Systemarchitektur auf drei Schichten abgebildet werden können. In der Abbildung 4.8 ist eine grafische Darstellung dieser Architektur zu sehen.

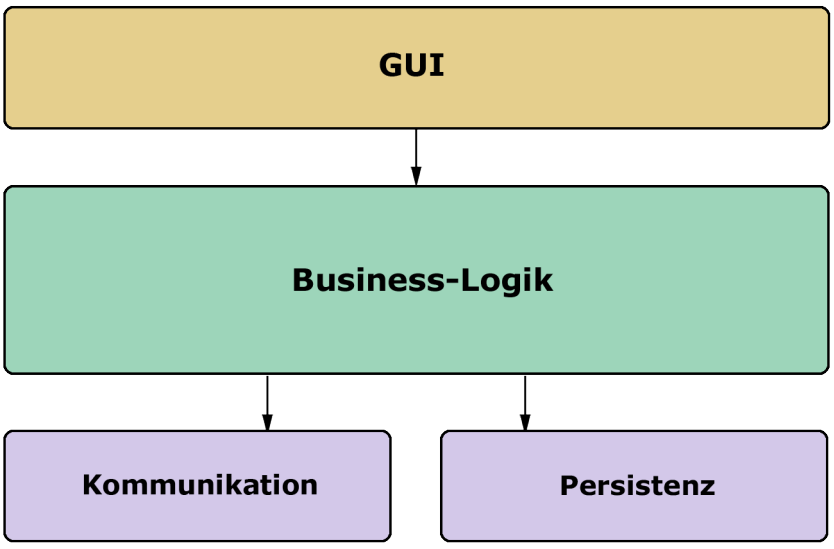

Abbildung 4.8: Architektur

Die obere Schicht bildet nach diesem Konzept die grafische Benutzer-Oberfläche der Anwendung (*GUI* in der Abbildung). Die *Business-Logik* beinhaltet die fachliche Funktionalität des Systems und stellt der GUI-Schicht eine Möglichkeit zur Verfügung, auf diese Funktionalität zuzugreifen. Die *Business-Logik* kann ihrerseits die Funktionalität der zwei Komponenten der untersten Schicht benutzen: *Kommunikation* und *Persistenz*. Diese Komponenten sind entsprechend für die Kommunikation zwischen den einzelnen Benutzern des Systems und für die Datenspeicherung verantwortlich.

4.3.1 Business-Logik

Den Kern der Architektur bildet die Geschäftslogik-Schicht (siehe „Business-Logik“ in der Abbildung). Anhand der in Abschnitt 3.2.2 beschriebenen Design-Anforderungen können folgende Module der Geschäftslogik definiert werden:

- Dokumentenerstellung;
- Gruppen-Verwaltung;
- Synchronisierung;
- Rechte-Verwaltung.

In der Abbildung 4.9 sind die einzelnen Komponenten des Business-Logik-Layers grafisch dargestellt.

Der darüber liegenden Schicht steht die Funktionalität dieses Layers über die Schnittstelle *BusinessManager* zur Verfügung. Dadurch wird für die obere Schicht eine Abstraktion von den konkreten Modulen dieses Layers erzeugt, die nach dem Fassaden-Entwurfsmuster [Gamma] gestaltet wurde.

Das Modul *Documents* repräsentiert das kollaborative Dokument für andere Komponenten des Systems und soll Objekte beinhalten, die seine Struktur und Daten abbilden. Dieses Modul übernimmt Aufgaben, die mit der Dokumentenerstellung verbunden sind. Dazu gehören in ersten Linie das Anlegen von neuen und das Öffnen von schon existierenden Dokumenten, das Zufügen von neuen Notizen an das zu bearbeitende Dokument und andere Operationen, die mit der Bearbeitung der kollaborativen Dokumente zusammenhängen. Da die Änderungen an einem Dokument in direkter Abhängigkeit von den Benutzer-Interaktionen an diesem Dokument stehen, hat der *Business-Manager* die Möglichkeit, die Funktionalität des Moduls direkt zu nutzen, um sie dann für das GUI zur Verfügung zu stellen.

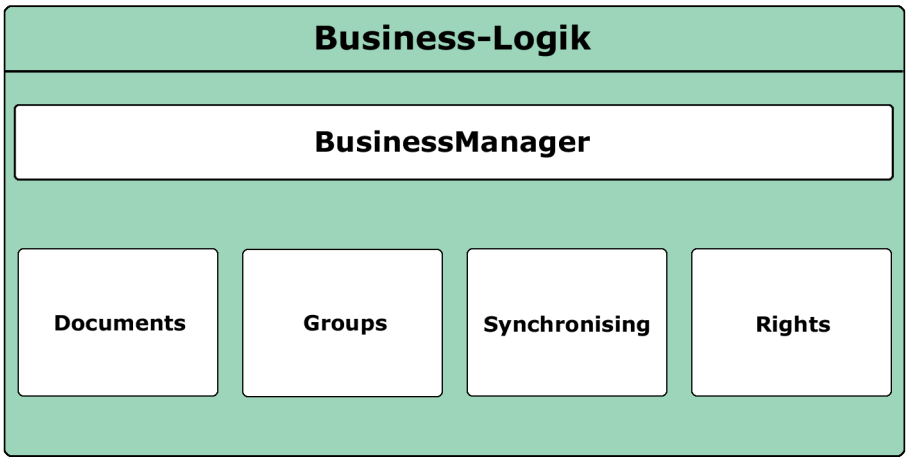

Abbildung 4.9: Architektur: Business-Logik-Schicht

Das Modul *Groups* übernimmt die Verwaltung von kollaborativen Gruppen und stellt für andere Module Objekte zur Verfügung, die ihnen den Zugriff auf die Gruppendaten ermöglichen. Zu den Aufgaben dieses Moduls zählen die Erstellung einer neuen Gruppe, das Verwalten von Mitgliedern einer Gruppe, die Suche nach neuen Gruppen und weitere Gruppen-Operationen. Wie auch beim *Documents*-Modul kann der *BusinessManager* auf die Funktionalität dieses Moduls direkt zugreifen, da sie für den Benutzer des Systems zur Verfügung stehen soll.

Das *Synchronising*-Modul ist für die Aufgaben verantwortlich, die mit dem Abgleich von zwei Versionen eines Dokumentes verbunden sind. Der Abgleich kann unter anderem durch den Benutzer initiiert werden. Aus diesem Grund hat der *BusinessManager* eine direkte Möglichkeit, seine Dienste zu nutzen. Das *Synchronising*-Modul benötigt darüber hinaus eine Zugriffsmöglichkeit auf das *Documents*-Modul, weil seine Aktivitäten direkte Auswirkung auf den Zustand der Objekte dieses Moduls haben können.

Des Modul *Rights* soll die Zugriffsrechte im System verwalten. Wie schon im Abschnitt 4.2 beschrieben wurde, wird das Thema der Zugriffsrechteverwaltung in dieser Masterarbeit nicht ausführlich behandelt. Es wird jedoch eine Schnittstelle vorgesehen, damit die Möglichkeit bestehen bleibt, die nötige Funktionalität nachzuentwickeln. Diese Schnittstelle wird durch den *Rights-Manager* beschrieben.

4.3.2 GUI

Die obere Schicht der Architektur bildet das grafische Benutzer-Interface (GUI) der Anwendung. Diese Komponente baut auf dem Logik-Layer auf, damit der Anwender Funktionen dieses Layers nutzen kann und über die kollaborativen Prozesse ständig informiert ist. Das GUI besteht aus zwei großen Modulen (siehe Abbildung 4.10). Das Modul *Kollaborativer Editor*

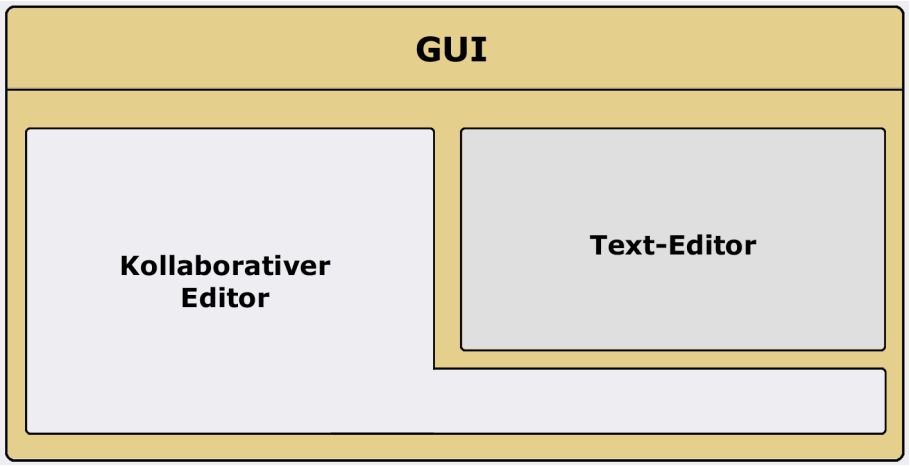

Abbildung 4.10: Architektur: GUI-Schicht

umfasst die eigene grafische Oberfläche des Systems. Durch dieses Interface steht dem Benutzer die Möglichkeit zur Verfügung, die kollaborativen Funktionen der Anwendung zu nutzen, wie z.B. das Anlegen der Struktur des Dokumentes, die Vergabe der Editierrechte etc. Da das Entwickeln eines umfangreichen Textbearbeitungsprogramms nicht zu den Zielen dieser Arbeit gehört, wird für das eigentliche Schreiben von Dokumenten ein beliebiger Text-Editor nach Wahl des Benutzers genommen, der das Format der zu bearbeitenden Dokumente unterstützt. Dieser Editor bildet das zweite Modul der Schicht und ist in der Abbildung als *Text-Editor* zu sehen. Die Funktionen zur Festlegung dieses Editors und für seine Aufrufe zum Editieren werden in das System integriert, um den kontinuierlichen Ablauf-Fluss der kollaborativen Prozesse zu gewährleisten.

Da in dieser Masterarbeit das LaTeX-Format für fachliche Inhalte des Dokumentes ausgewählt wurde, soll für das Schreiben der Dokumente ein beliebiger LaTeX-Editor eingesetzt werden.

4.3.3 Persistenz-Komponente

Die *Persistenz*-Komponente übernimmt die dauerhafte Speicherung der Dokumente, sowohl ihrer kollaborativen Informationen als auch der fachlichen Inhalte, und der kollaborativen Gruppen. Darüber hinaus ist dieses Modul für die Abbildung der Persistenz-Daten auf Objekte, mit denen andere Komponenten arbeiten, und umgekehrt verantwortlich. Für die *Business-Logik*-Schicht bietet das Persistenz-Modul die Schnittstelle *PersistenceManager*, über die auf ihre Funktionalität zugegriffen werden kann. Zu den Aufgaben dieser Schnittstelle zählt in erster Linie das Weiterleiten der Aufrufe der anderen Komponenten an die entsprechenden funktionalen Module dieses Layers. Sie bildet damit eine Fassade dieser Komponente. So bleibt die eigentliche Persistenz-Logik nach außen verborgen, wodurch ein hoher Flexibilitätsgrad in Bezug auf die Gestaltung dieses Layers erreicht wird.

4.3.4 Kommunikations-Komponente

Die letzte Komponente der Architektur bildet das *Kommunikation*-Modul. Dieses Modul ist für die Peer-to-Peer-Kommunikation zwischen den Gruppenmitgliedern verantwortlich. Es übernimmt den Aufbau der Verbindungen zu den anderen Teilnehmern, die Aufrechterhaltung der Kommunikationskanäle und die Übertragung der Daten zwischen den Benutzern des Systems. Seine Dienste werden den Modulen der darüber liegenden Business-Logik Schicht über die Schnittstelle *CommunicationManager* zur Verfügung gestellt.

4.3.5 Zusammenspiel der einzelnen Schichten

Das Zusammenspiel der *GUI*-Schicht und der *Business-Logik*-Schicht basiert auf der Kombination aus dem etablierten Model-View-Controller-Paradigma [Gamma] und dem Fassade-Pattern [Gamma]. Eine schematische Darstellung der entsprechenden Beziehungen ist im Klassendiagramm 4.11 zu sehen.

Das Model-View-Controller-Prinzip gestattet die Entkopplung der Benutzeroberfläche von der Business-Logik und dem fachlichen Modell und ermöglicht bei Bedarf einen leichten Austausch der GUI-Komponenten. Das Modell bilden zwei Module der Architektur: *Documents* und *Groups*, da sie unter anderem die Fach-Objekte des Systems (Dokument und Gruppe) repräsentieren. Diese Module sind im Diagramm durch *DocumentManager* und *GroupManager* dargestellt. Der *CollaborativeEditor* entspricht in der Archi-

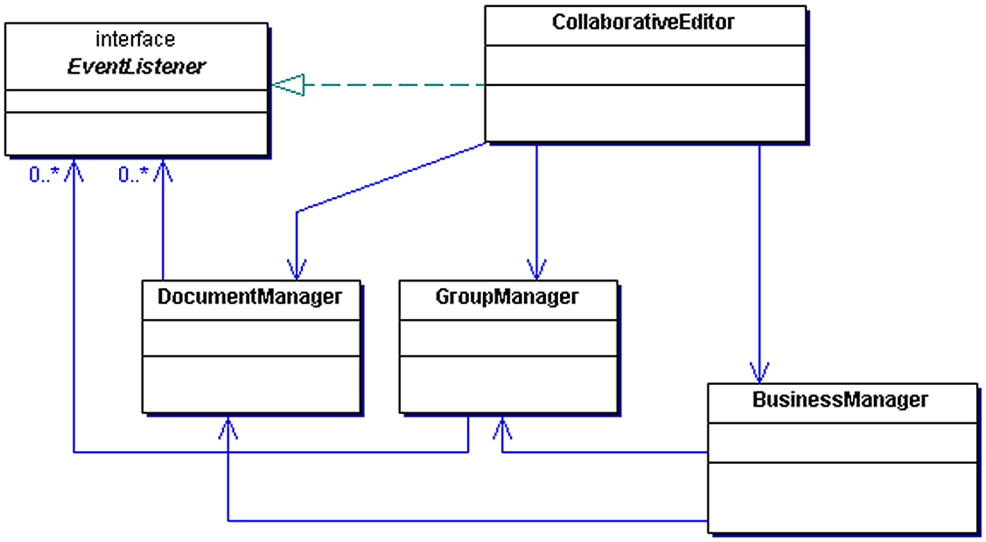

Abbildung 4.11: Beziehungen zwischen GUI und Business-Logik

tektur der Kombination aus dem View und dem Controller. Er präsentiert die Daten für den Benutzer, wertet die Eingaben des Benutzers aus und initiiert abhängig davon die entsprechenden Aktionen der Geschäftslogik. Dabei greift der *CollaborativeEditor* auf die Fassade dieser Schicht zu, den *BusinessManager* (siehe Diagramm). Zu den Aufgaben des *BusinessManager* gehört das Weiterleiten der initiierten Aktionen an die dafür verantwortlichen fachlichen Module dieser Schicht.

Entsprechend diesen Zusammenhängen referenziert der *CollaborativeEditor* den *BusinessManager*. Der *DocumentManager* und der *GroupManager* werden sowohl von dem *CollaborativeEditor* als auch von dem *BusinessManager* referenziert. Die lose Kopplung zwischen dem Modell und dem View wird durch das Beobachter-Entwurfsmuster realisiert [Gamma]. Demzufolge implementiert der *CollaborativeEditor* das *EventListener*-Interface (siehe Diagramm). Jedes *EventListener*-Objekt kann sich bei dem *DocumentManager* und bei dem *GroupManager* eintragen, damit es bei jeder Änderung des Modells darüber informiert wird und die Darstellung der Modell-Objekte entsprechend anpassen kann.

Die Beziehungen zwischen der *Business-Logik*-Schicht und den Komponenten *Kommunikation* und *Persistenz* werden nach dem Fassade-Pattern [Gamma] umgesetzt, um die Flexibilität des Systems in Bezug auf das Design dieser zwei Komponenten zu erhöhen. Dementsprechend können die Business-Logik-

Module nur über definierte Schnittstellen auf die Funktionalität dieser Komponenten zugreifen.

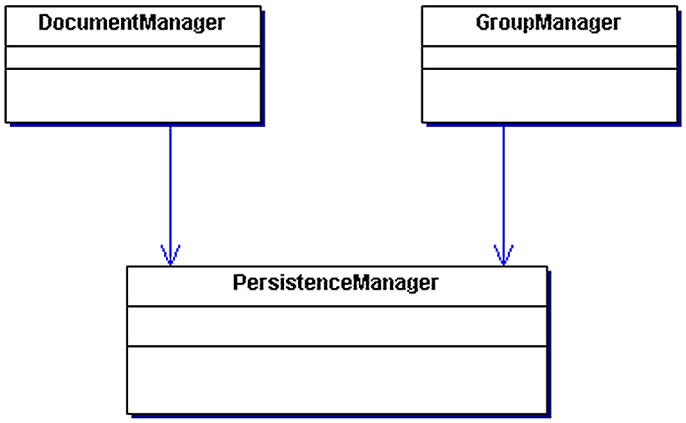

Abbildung 4.12: Beziehungen zwischen Business-Logik und der Persistenz-Komponente

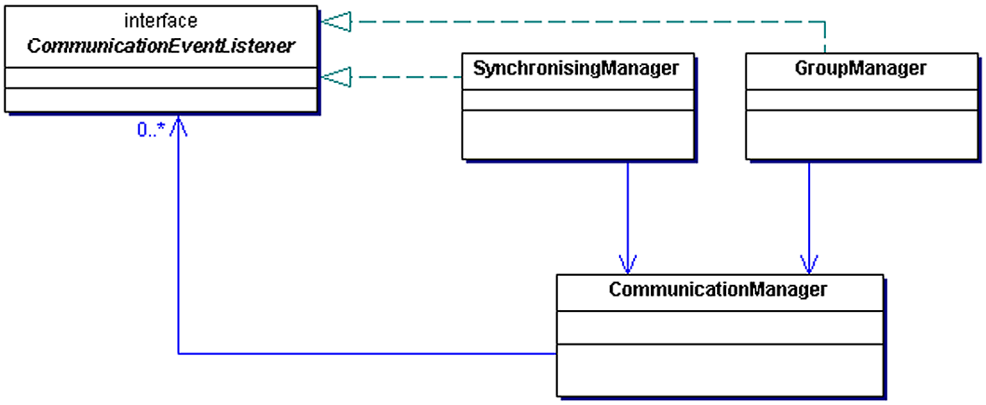

Abbildung 4.13: Beziehungen zwischen Business-Logik und der Kommunikations-Komponente

Die Funktionalität der Persistenz-Komponente wird von den Modulen *Documents* und *Groups* benötigt. Der Zugriff auf diese Komponente erfolgt, wenn ein Dokument- oder Gruppen-Objekt geladen werden soll oder wenn die Änderungen an diesen Objekten dauerhaft gespeichert werden müssen (siehe Abbildung 4.12).

Die Funktionalität der Kommunikations-Komponente wird von den Modulen *Synchronising* und *Groups* verwendet, da für die Abläufe dieser Module die Kommunikation mit den anderen Team-Mitgliedern erforderlich ist. Der Austausch von Nachrichten zwischen den System-Nutzern soll asynchron erfolgen, um ein unnötiges Warten auf eine Antwort zu vermeiden. Aus diesem Grund werden das *Groups*- und das *Synchronising*-Modul als Beobachter der Kommunikations-Komponente agieren. Dementsprechend müssen sie das Interface *CommunicationEventListener* implementieren. Im Klassendiagramm 4.13 sind die entsprechenden Zusammenhänge zu sehen.

4.3.6 Dynamisches Zusammenspiel der einzelnen Module

Das dynamische Zusammenspiel der Module wird am Beispiel von zwei Anwendungsfällen präsentiert.

Der erste Anwendungsfall beschreibt das Erstellen eines neuen Dokumentes durch einen Benutzer (siehe „Analyse der Aufgabe“, Abschnitt 3.2.2). An diesem Vorgang sind drei Module des Systems beteiligt: *CollaborativeEditor*, *Documents* und *Persistence*-Modul. Darüber hinaus nimmt der *Business-*

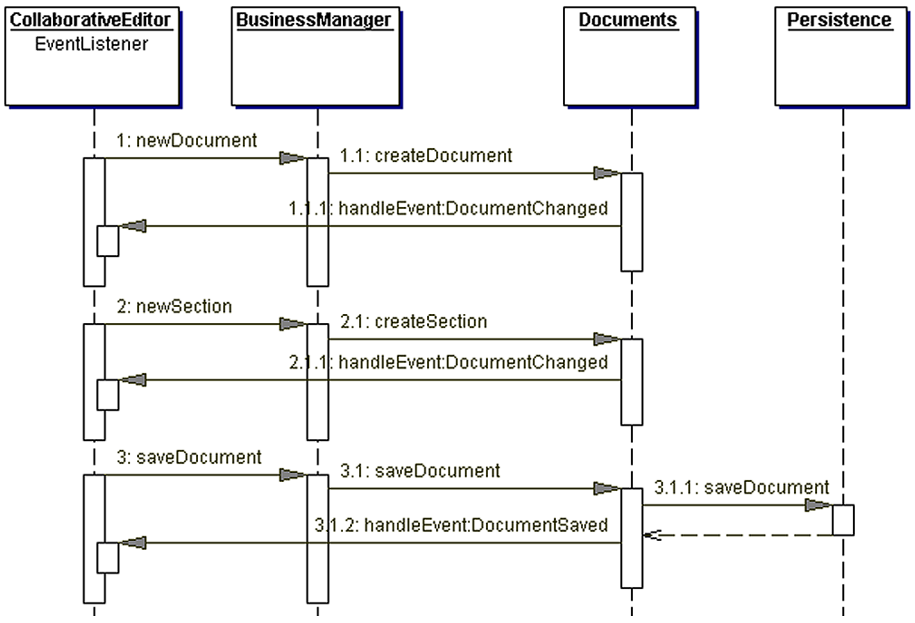

Abbildung 4.14: Zusammenspiel der Module: Erstellen eines neuen Dokumentes

Manager an diesem Geschäftsprozess teil. Der Anwendungsfall beinhaltet drei wesentliche Schritte: Anlegen des neuen Dokumentes, die Bildung seiner vorläufigen Struktur und Speichern des neuen Dokumentes. Alle diese Schritte werden von dem Benutzer durchgeführt. Der Gesamtablauf ist im Sequenzdiagramm 4.14 zu sehen. Das Anlegen des Dokumentes ist in diesem Diagramm durch die Operation *newDocument* dargestellt. Wenn das Dokument erfolgreich erstellt wird, wird der *CollaborativeEditor* darüber durch ein Event benachrichtigt, damit er seine Dokument-Darstellung aktualisieren kann. Die Bildung der Struktur des Dokumentes ist im Diagramm durch die Operation *newSection* präsentiert. Das Dokument soll beispielhaft vorerst aus einem Kapitel bestehen. Nach dieser erfolgreichen Operation wird der *CollaborativeEditor* ebenfalls darüber informiert. Die letzte Operation hat als Ziel das Speichern des neuen Dokumentes. Sie ist im Diagramm als *saveDocument* zu sehen.

Der zweite Anwendungsfall stellt die Suche nach neuen kollaborativen Gruppen im Netz dar. An diesem Prozess nehmen folgende Module teil: *CollaborativeEditor*, *Groups*-Modul und *Communication*-Modul. Der *BusinessManager* ist als die Schnittstelle zur Business-Logik ebenfalls daran beteiligt. Der Ablauf des Vorgangs ist im Diagramm 4.15 zu sehen. Die Su-

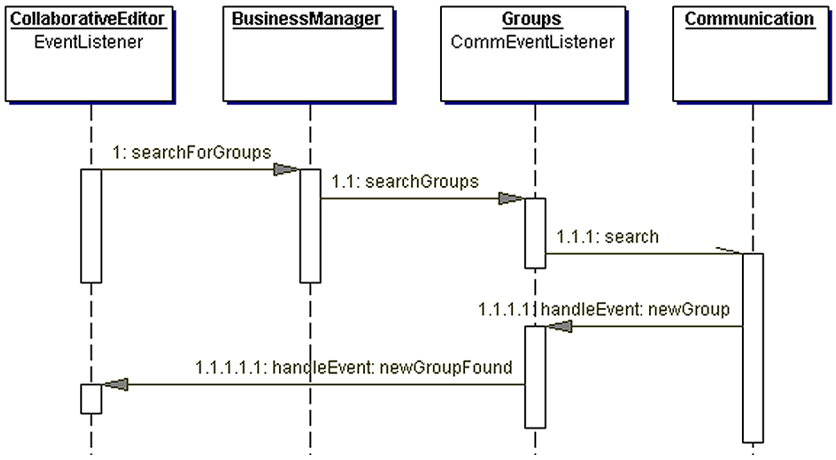

Abbildung 4.15: Zusammenspiel der Module: Suche nach Gruppen

che wird durch den Benutzer initiiert und von dem *CollaborativeEditor* an den *BusinessManager* weitergeleitet. Der gibt diese Aufgabe an das *Groups*-Modul (Operation *searchGroups*) weiter. Die eigentliche Suche wird von dem Kommunikations-Modul durchgeführt. Wenn eine neue Gruppe im Netz gefunden wurde, wird das *Groups*-Modul darüber durch ein Kommunikations-

Event informiert. Dieses Modul benachrichtigt seinerseits den *CollaborativeEditor*, wodurch er seine Präsentation aktualisiert und dem Benutzer die gefundenen Gruppen anzeigt.

4.4 Entwurf der Module

In den weiteren Abschnitten wird der Entwurf der einzelnen Layer der Architektur beschrieben.

4.4.1 Das Dokument-Modell

Wie in den vorherigen Abschnitten erläutert wurde, ist das *Documents*-Modul für die Durchführung der Operationen auf dem kollaborativen Dokument und zu seiner Präsentation für andere Module des Systems verantwortlich. Dementsprechend soll dieses Modul das Modell-Objekt des zu bearbeitenden Dokumentes beinhalten. Da in dieser Arbeit kollaborative Dokumente ein XML-Format haben werden, erscheint es sinnvoll, für ihre Repräsentation ein XML-Objekt einzusetzen. Das Dokument wird nach außen aber nicht durch dieses Objekt präsentiert, sondern durch die Klasse *DocumentManager*. Dieser Abstraktionsschritt ist nötig, weil das kollaborative System die Bearbeitung einer beliebigen Anzahl von Dokumenten gestattet, zu einem Zeitpunkt aber nur eines. Dadurch kann nicht zu jedem Zeitpunkt ein Dokument-Objekt existieren (z.B. vor dem Öffnen eines Dokumentes oder nach dem Schließen des aktuellen Dokumentes). Das Modell-Objekt muss aber zur Laufzeit immer vorhanden sein, da andere Module es referenzieren (siehe Abschnitt 4.3.5). Darüber hinaus kapselt der *DocumentManager* die Zugriffe auf das XML-Objekt und die damit verbundene Navigation auf seiner Baum-Struktur ein, so dass für andere Module eine wohl definierte Schnittstelle zur Verfügung gestellt wird. So können andere System-Komponenten Informationen über den Titel des Dokumentes, seine Abschnitte, Annotationen eines Abschnitts usw. bekommen, ohne Kenntnisse über seinen internen Aufbau haben zu müssen.

Das Modul soll folgende Funktionalität anbieten:

- Anlegen neuer Dokumente;
- Erstellen neuer Abschnitte im Dokument;
- Zufügen und Verwalten von Annotationen;

- Abschließen des Dokumentes oder seiner Teile;
- Löschen von Abschnitten des Dokumentes;
- Öffnen der Dokumentabschnitte zum Editieren (Laden der fachlichen Dokumenteninhalte);
- Laden und Speichern der Dokumente.

Darüber hinaus soll eine weitere Funktion umgesetzt werden, die, entsprechend dem im vorigen Abschnitt beschriebenen Beobachter-Entwurfsmuster, erlaubt, dem Dokument-Modell Beobachter hinzuzufügen und sie über die Änderungen der Modell-Daten zu benachrichtigen.

Des Weiteren spielt die Integrität der Daten, wie in jedem System, eine wichtige Rolle. Dieser Aspekt der Sicherheit ist besonders bei dem Öffnen der Dokumente in einem externen Editor von Bedeutung, da entsprechend der Zugriffsberechtigungs-Strategie alle Gruppenmitglieder die Dokument-Inhalte ansehen können. Dabei verliert das Editiersystem die Kontrolle über die Handlungen des Benutzers. Um sicherzustellen, dass keine unerlaubten Veränderungen am Dokument vorgenommen werden, muss ein passender Mechanismus ausgearbeitet werden. Z.B. könnte vor dem Öffnen des Dokumententeils eine Sicherheitskopie seines Inhaltes erstellt werden. Des Weiteren könnte ein Hash-Code-Verfahren angewendet werden, für die Sicherstellung, ob Veränderungen gemacht wurden. Da das Thema der Sicherheit nicht zu den Schwerpunkten dieser Arbeit gehört und bei dem Entwurf von einer gutwilligen Umgebung ausgegangen wird, wird auf die Ausarbeitung einer dafür geeigneten Strategie verzichtet.

4.4.2 Die Persistenz-Komponente

Die Persistenz-Komponente ist für das unmittelbare Speichern und Laden der kollaborativen Dokumente und Gruppen verantwortlich. Wie schon in vorherigen Abschnitten beschrieben wurde, besteht das kollaborative Dokument aus drei logischen Teilen (siehe Abschnitt „Logische Aufteilung des Dokumentes“ auf Seite 63). Die kollaborativen Informationen werden in einem XML-Format repräsentiert. Seine Metadaten und fachliche Inhalte werden im LaTeX-Format gespeichert. Die Aufgabe besteht also darin, eine Strategie zu entwerfen, wie die Daten, sowohl XML- als auch LaTeX-Inhalte, persistent gemacht werden können. Die Persistenz-Haltung von Gruppen wird dementsprechend auch auf der gewählten Strategie basieren.

Es gibt zwei grundlegende Möglichkeiten, Daten zu speichern: in einer Datenbank oder Dateisystem-basiert. Da die unterschiedlichen Teile des gesamten kollaborativen Dokumentes auf zwei verschiedene Formate abgebildet werden, lohnt es sich, die Speicherungsalternativen für beide Formate getrennt zu untersuchen.

4.4.2.1 Datenbank-basierter Ansatz

Für die Speicherung der LaTeX-Dokumente in einer Datenbank kommen zwei Möglichkeiten in Frage: als ein Binary Large Object (BLOB)[4] oder als ein Character Large Object (CLOB)[5] in einer relationalen Datenbank-Tabelle. Für die XML-Strukturen existieren dagegen mehrere Datenbank-basierte Alternativen, die hier kurz erläutert werden. In relationalen Datenbankmanagementmystemen können XML-Strukturen, analog zu LaTeX-Dokumenten, als binäre oder textuelle Daten vollständig in einer Spalte einer Tabelle gespeichert werden. Bei diesem Ansatz werden aber keine Vorteile aus der Strukturierung der Daten gewonnen. So muss z.B. bei Änderungen das ganze Dokument gesperrt werden und eine effektive Suche kann nur durch den Einsatz von aufwändigen Volltext-Indizierungen realisiert werden [Wiese]. Bessere Lösung bietet ein weiterer Ansatz, bei dem hierarchische XML-Strukturen auf ein festes Datenbank-Schema und auf relationale Datenbankstrukturen abgebildet werden [Wiese], was durch einen zusätzlichen Umwandlungsprozess möglich wird. Der Nachteil dieser Lösung liegt darin, dass durch den hierarchischen Aufbau der Dokumente die Tabellen mit sich selbst verknüpft werden müssen und dadurch rekursive Beziehungen entstehen [Wiese].

Abgesehen von relationalen Datenbanken gibt es eine weitere Alternative, XML-Dokumente DB-basiert zu speichern. Diese Alternative liegt im Einsatz eines nativen XML-Datenbankmanagementsystems, das XML-Dokumente im Ganzen, ohne Konvertierungen, speichert [Wiese]. Die Dokumentordnung und seine Strukturierung bleiben dabei erhalten. Solche DBMS unterstützen XML-Schemata, mit deren Hilfe eine bessere Konsistenz und das Validieren der Daten vor dem Import realisierbar ist. Darüber hinaus ermöglichen sie durch den Einsatz von speziellen XML-Anfrage-Sprachen wie XQuery oder XPath schnellen und effizienten Zugriff auf die Daten. Als Nachteil des Einsatzes von XML-Datenbanksystemen kann der Aspekt genannt werden, dass XML-DBMS nicht so ausgereift wie die relationalen Datenbanksysteme sind, da sie relativ jung sind.

[4]Ein BLOB ist ein Datenbankfeld zur Speicherung von sehr großen binären Daten.
[5]Ein CLOB ist ein Datenbankfeld zur Speicherung von sehr langen Textdaten.

Datenbankmanagementsysteme bieten im Allgemeinen ein Paket von Standard-Services, die viele Vorteile für die Bearbeitung von Daten mit sich bringen. Dazu gehören solche Dienste wie Mehrbenutzerfähigkeit (Datenschutz, Sperren, Daten-Synchronisation), Datensicherung (Datenbank-Backups), Datenintegrität (Transaktionen), effizienter Zugriff auf die Daten (Indizierung, Anfrage-Optimierung) etc. [Kemper]. Da das in dieser Masterarbeit zu entwickelnde kollaborative Editiersystem als ein Peer-to-Peer-System konzipiert wird, können in diesem System entweder lokale Datenbanken, für jede laufende Applikation eine, oder verteilte Datenbanken mit Replikation und ohne Fragmentierung, da jeder Gruppenmitglied das ganze Dokument bei sich lokal haben soll, eingesetzt werden. Im Fall der verteilten Datenbanken steht die Wahl eines passenden Replikationsverfahrens im Vordergrund des Datenbank-Designs. Da die einzelnen Applikationen sich häufig im offline-Modus befinden werden und ein zentraler Server im System nicht existiert, werden an das Replikationsverfahren anspruchsvolle Anforderungen gestellt.

4.4.2.2 Datei-basierter Ansatz

Im Vergleich zu DBMS besitzt ein Datei-basierter Ansatz eine flache Strukturierung auf der Datei-Ebene. Obwohl die Verwaltung der Daten in einem Dateisystem sich durch wenige der DBMS-Eigenschaften auszeichnet, weist sie einige andere Vorteile auf. So z.B. kostet dieser Ansatz weniger Ressourcen als ein Datenbankmanagementsystem. Außerdem gibt es bei einem Dateisystem keinen zusätzlichen Installationsaufwand. Darüber hinaus erscheint es in dem zu entwickelnden Editiersystem sinnvoll, ein kollaboratives XML-Dokument komplett und nicht teilweise für die Bearbeitung zu laden, da nur ein ganzes Dokument dem Benutzer maximale Informationen über die Struktur des Dokumentes und kollaborative Prozesse an diesem Dokument vermitteln kann. Da das XML-Dokument im System als ein XML-Objekt repräsentiert werden kann und für die Navigation auf seinem Baum eine standardisierte XML-Anfragesprache, wie z.B. XPath, verwendet werden kann, unterscheidet sich der Datei-basierte Ansatz in der Effizienz der Zugriffe nicht von XML-Datenbankmanagementsystemen.

Da das Thema der Persistenz nicht zu den Schwerpunkten dieser Arbeit gehört, wird für die Speicherung der XML-Dokumente der einfachere Datei-basierte Ansatz gewählt, obwohl die XML-DBMS eine interessante Alternative dazu bieten. Die Gruppendaten werden dementsprechend auch in Dateien gespeichert. Um den Entwurf für andere Persistenz-Lösungen trotzdem fle-

xibel zu halten, wird von der tatsächlichen Speicherungsmethode durch eine Schnittstelle abstrahiert, die die für Speichern und Laden der Dokumente erforderliche Funktionalität beschreiben wird.

4.4.2.3 Design des Moduls

Wie es schon beschrieben wurde, sind Dienste des Persistenz-Moduls für andere System-Komponenten über ihre Fassade *PersistenceManager* erreichbar. Das Modul bietet folgende grundlegende Funktionalität:

- Laden eines Dokumentes;
- Laden des entsprechenden fachlichen Inhaltes eines der Abschnitte des Dokumentes;
- Laden der Titel aller vorhandenen Dokumente;
- Speichern eines Dokumentes;
- Export der Dokument-Inhalte;
- Speichern und Laden der Gruppen-Informationen.

Der *PersistenceManager* leitet die Aufgaben nach Bedarf an zwei weitere funktionale Komponenten des Moduls weiter.

Die erste Komponente ist für das eigentliche Speichern und Laden der Daten verantwortlich. Unter dem Laden eines Dokumentes wird das Laden seiner kollaborativen Informationen und Mapping dieser Informationen auf das entsprechende XML-Objekt verstanden. Die fachlichen Inhalte werden grundsätzlich erst bei Nachfrage geladen. Dabei werden darunter alle Schritte verstanden, die nötig sind, um einen gespeicherten Inhalt zum Öffnen in einem Text-Editor vorzubereiten, so z.B. Schreiben des Inhaltes in eine temporäre Datei. Das Laden der Titel aller vorhandenen Dokumente und das Laden der Gruppennamen wird z.B. zur Auswahl eines bestimmten Dokumentes bzw. einer bestimmten Gruppe durch den Benutzer verwendet. Um das Design des Moduls flexibel in Bezug auf die verwendete Persistenz-Methode zu gestalten, wird hier das Strategie-Pattern [Gamma] angewendet. Dementsprechend wird die abstrakte Funktionalität der Komponente durch die Schnittstelle *PersistenceIf* definiert. In Abhängigkeit von dem ausgewählten Datenhaltungs-Ansatz kann diese Funktionalität konkret umgesetzt werden. In dieser Arbeit wird diese Aufgabe durch die Klasse *PersistenceMaker* übernommen, die eine Datei-basierte Datenhaltung implementiert.

Die zweite Komponente ist für die Operationen mit dem Format verantwortlich, in dem die fachlichen Inhalte der kollaborativen Dokumente gespeichert werden. Im Fall dieser Arbeit ist es das LaTeX-Format. Um die Flexibilität dieses Moduls in Bezug auf das ausgewählte Dokumentenformat zu gewährleisten und das System offen für andere Formate zu lassen, wird das Design dieser Komponente analog zu der ersten Komponente des Persistenz-Moduls nach dem Strategie-Pattern gestaltet [Gamma]. Dementsprechend wird ihre Funktionalität in Form des Interfaces *CreatorIf* definiert. Diese Funktionalität kann nach Bedarf durch verschiedene konkrete Format-Klassen entsprechend dem ausgewählten Format implementiert werden. In dieser Arbeit übernimmt diese Aufgabe die Klasse *LatexCreator*.

In der Abbildung 4.16 sind das Klassendiagramm der Persistenz-Komponente und statische Abhängigkeiten ihrer einzelnen Bestandteile untereinander zu sehen.

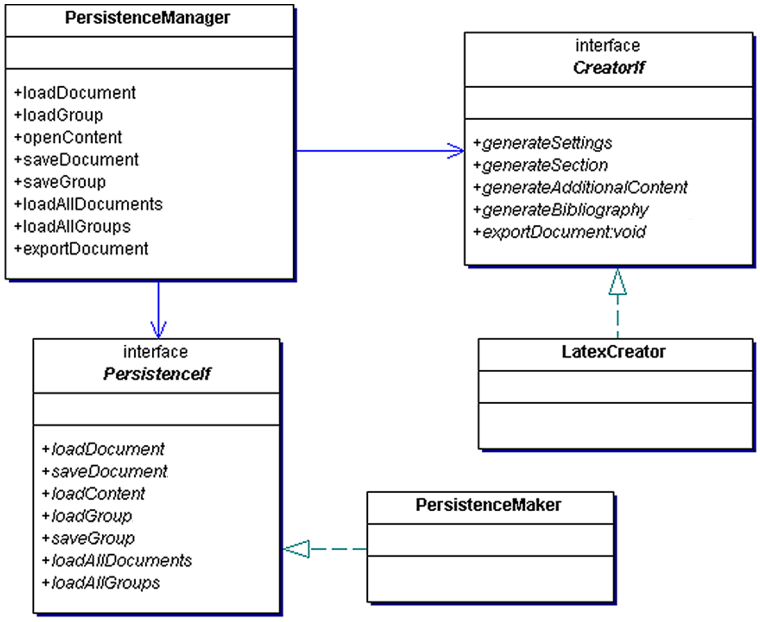

Abbildung 4.16: Persistenz: Klassendiagramm

Die *Creator*-Komponente kommt beim Öffnen eines Teils des Dokumentes oder beim Speichern des Dokumentes zum Einsatz, aber nur dann, wenn die benötigten Strukturen noch nicht existieren oder aktualisiert werden müssen. Daher gehört zu den Aufgaben des Moduls das Generieren der Standard-Strukturen eines Dokumentes. Dazu zählen:

- Generieren der Standard-Einstellungen für ein Dokument. Darunter werden Informationen verstanden, die das Aussehen des Dokumentes bestimmen. Hier können zusätzliche Formatierungspakete eingebunden werden, die für die Gestaltung des Dokumentes benötigt werden.

- Generieren eines fachlichen Abschnitts. Die Bezeichnung eines Abschnitts ist in Dokumentenformaten sehr oft von seiner Hierarchie-Ebene abhängig. So heißen z.B. in LaTeX die Abschnitte *chapter*, *section*, *subsection* usw. Daher erscheint es sinnvoll, die Erstellung des Standard-Inhaltes für einen Abschnitt von seinem Hierarchie-Level im Dokument abhängig zu machen. Darüber hinaus ist die Hierarchie-Tiefe im LaTeX-Format begrenzt. Dementsprechend müssen für das Generieren eines fachlichen Inhaltes entsprechende Mechanismen eingebaut werden, die kontrollieren, dass die erlaubte Tiefe nicht überschritten wird.

- Generieren eines zusätzlichen Inhaltes, wie z.B. eines Literaturverzeichnisses.

- Zusammenstellen der Meta-Informationen eines Dokumentes, die beschreiben, aus welchen Teilen das Dokument zusammengesetzt wird. Daraus soll das Meta-Dokument entstehen, in dem der Bezug auf alle anderen Inhalte (Einstellungen, einzelne Abschnitte usw.) durch die entsprechenden Verweise erzeugt wird. Diese Meta-Daten werden für die Bearbeitung des Dokumentes außerhalb des kollaborativen Systems benötigt, z.B. für die Erstellung aus dem Dokument einer PDF-Datei. Das Generieren der Meta-Informationen soll bei jedem Speichern des Dokumentes durchgeführt werden, um so die Konsistenz des Inhaltes zu garantieren.

- Export der fachlichen Inhalte. Darunter wird der Export aller Abschnitte des Dokumentes, seiner Einstellungen und des Meta-Dokumentes verstanden, der das Dokument zur Bearbeitung außerhalb des kollaborativen Systems möglich macht, z.B. zur Generierung der Endfassung im PDF-Format.

In der Abbildung 4.17 ist das dynamische Zusammenspiel der Objekte des Persistenz-Moduls am Beispiel des Speicherns eines Dokumentes dargestellt worden. Demnach wird zuerst von dem *PersistenceManager* überprüft, ob für alle Teile des Dokumentes die entsprechenden fachlichen Inhalte existieren (Einstellungen, logische Abschnitte des Dokumentes und zusätzliche Inhalte). Wenn das nicht der Fall ist, werden die fehlenden Strukturen durch den

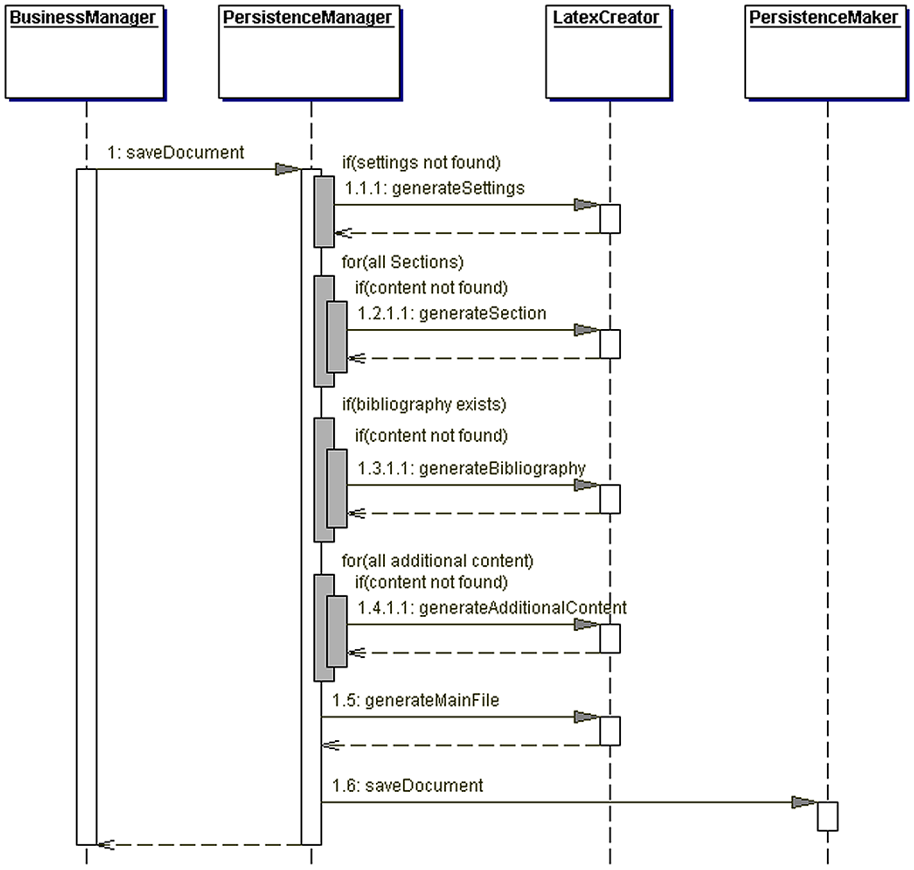

Abbildung 4.17: Sequenzdiagramm: Speichern des Dokumentes

LatexCreator erstellt. Danach wird das Meta-Dokument aktualisiert (siehe die Operation *generateMainFile* im Diagramm). Als letzter Schritt wird das Dokument durch den *PersistenceMaker* gespeichert.

4.4.3 Das Synchronisierungsmodul

Das Synchronisierungsmodul ist für den Abgleich und die Aktualisierung der Dokument-Versionen verantwortlich, die bei verschiedenen Gruppenmitgliedern existieren. Jedes Mal, wenn ein Teilnehmer wieder in die Gruppe eintritt (wieder online ist), kann die Synchronisierung seines Dokumentes mit den Versionen der anderen Teilnehmer durchgeführt werden. Darüber hinaus kann die Synchronisierung auch durch den Benutzer selbst zu einem beliebigen Zeitpunkt initiiert werden. Das Modul kann in der Rolle sowohl

des Initiators der Synchronisierung als auch des zweiten Beteiligten auftreten, abhängig davon, in welchem Zustand sich die Applikation befindet. Das Modul wird durch die Klasse *SynchronisingManager* repräsentiert.

Die Synchronisierung soll aus der Sicht der initiierenden Seite in fünf Schritten ablaufen. Im ersten Schritt werden aktuelle Versionsnummern aller Teile des entfernten Dokumentes angefordert. Als Antwort darauf wird ein bestimmtes XML-Objekt erwartet, das diese Informationen enthält. Das XML-Schema für dieses Objekt ist in der Abbildung 4.18 zu sehen. Demzufolge

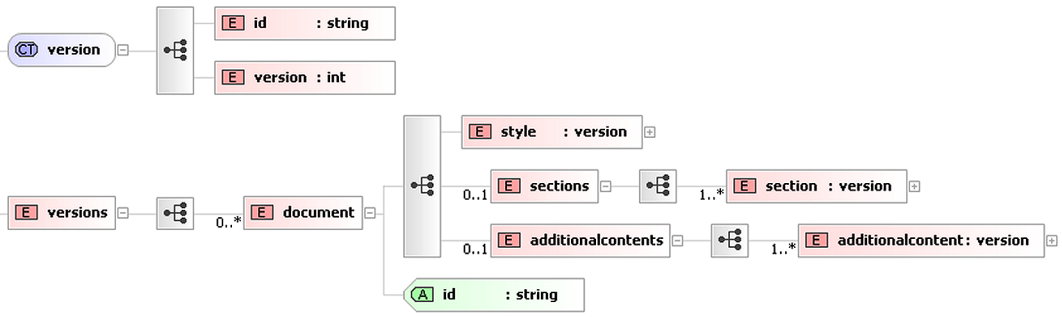

Abbildung 4.18: Das Versions-Schema

kann jedes Versions-Objekt Informationen über mehrere Dokumente enthalten, da zu einer kollaborativen Gruppe mehrere Dokumente gehören können. Jedes Dokument-Objekt wird durch seine Id eindeutig identifiziert. Außerdem enthält dieses Objekt weitere Elemente, die die Version der Einstellungen (*style*), Versionen der einzelnen Abschnitte des Dokumentes (*section*) und Versionen der zusätzlichen Inhalte (*additionalcontent*), wenn solche existieren, beschreiben. Für diese Elemente wurde ein spezieller XML-Typ definiert, der zwei Elemente beinhaltet: die Id eines Dokumententeils und die dazugehörige Versionsnummer (siehe *version* in der Abbildung).

Sobald das Versions-Objekt erhalten wird, werden die lokalen Versionen des Dokumentes mit den Remote-Versionen verglichen. Anhand dieses Vergleichs wird eine Liste mit den Ids der Dokumententeile erstellt, deren lokale Versionsnummer kleiner ist als in dem entfernten Dokument oder die lokal noch nicht existieren. Diese Liste wird durch ein weiteres XML-Objekt repräsentiert und an den entfernten Kommunikationspartner übergeben. Das XML-Schema zu diesem Objekt ist in der Abbildung 4.19 dargestellt. Wie auch das Versions-Objekt kann das Identifikatoren-Objekt mehrere Dokument-Elemente beinhalten, die durch ihre Id gekennzeichnet werden (siehe Abbildung). Das Dokument-Element kann, identisch zum Versions-Objekt, drei

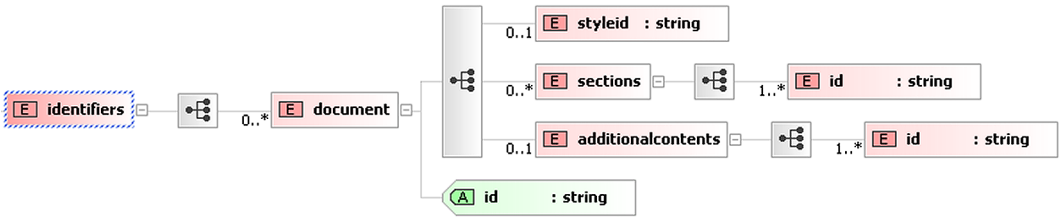

Abbildung 4.19: Das Identifikatoren-Schema

Datenbereiche umfassen: für die Einstellungen, für die Abschnitte des Dokumentes und für die zusätzliche Inhalte. Jeder dieser Bereiche enthält die vorher bestimmten Ids.

Als Antwort bekommt das Synchronisierungsmodul die aktuellen Daten der angeforderten Dokumententeile. Im weiteren Schritt wird das lokale Dokument aktualisiert, indem die entsprechenden Elemente seines Modells angepasst und die dazugehörigen fachlichen Inhalte mit den neuen Versionen überschrieben werden. Als letztes wird das Dokument gespeichert.

Der komplette Ablauf aus der Sicht des Initiators der Synchronisierung ist im Sequenzdiagramm 4.20 zu sehen.

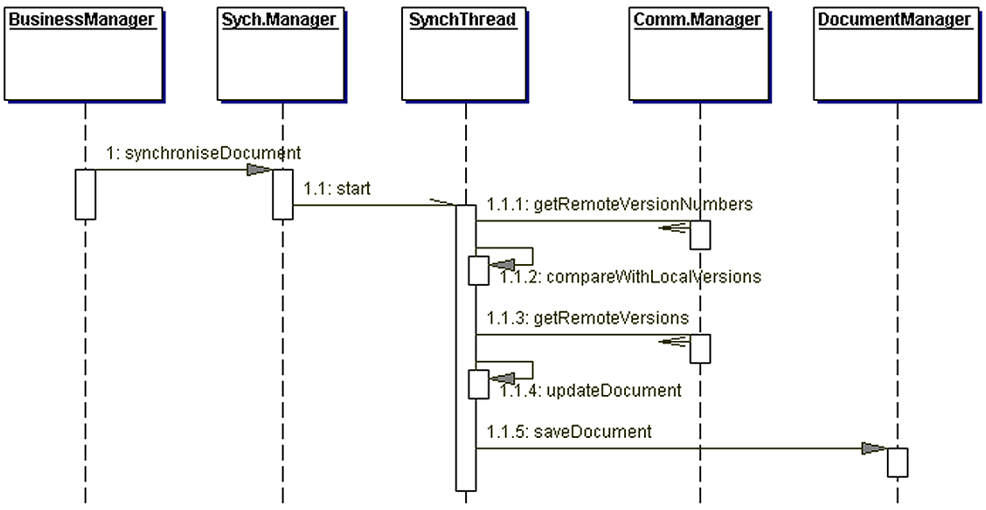

Abbildung 4.20: Sequenzdiagramm: Synchronisierung des Dokumentes

Wenn ein Teilnehmer seine kollaborative Gruppe wieder betritt und ein

potenzieller Synchronisierungspartner entdeckt wurde, startet der Abgleich nicht gleich, sondern erst nach der Bestätigung des Benutzers, nachdem er darüber informiert wurde. Dieser zusätzliche Schritt ist nötig, da dadurch das Benutzer-Bewusstsein über die im System ablaufenden Prozesse gesteigert wird. Außerdem wird dem Benutzer das Recht vorbehalten bleiben, selbst zu entscheiden, ob die Aktualisierung durchgeführt werden soll oder nicht. Besonders wichtig ist es in einem System, in dem ein Konzept für die Versionskontrolle nicht unterstützt wird.

Da die Synchronisierung einige Zeit beanspruchen kann, findet sie asynchron statt, in einem extra Thread, um keine andere Prozesse zu beeinträchtigen. Es kann vorkommen, dass ein System-Teilnehmer mehrere Synchronisierungswünsche nacheinander erhält, wenn z.B. mehrere Team-Mitglieder gleichzeitig die Gruppe betreten. Damit es nicht zu Inkonsistenzen kommt und die ankommenden Anfragen nicht verloren gehen, werden sie in einer Warteschlange verwaltet.

Die Dokumente werden zwischen zwei Synchronisierungspartnern im gleichen Format ausgetauscht, in dem das Dokument lokal existiert. Durch diese Vorgehensweise können die veralteten kollaborativen Inhalte leicht durch neue Versionen ersetzt werden. Bei der Zusammenführung von zwei Versionen eines Dokumentes können aufgrund der im System verwendeten Editierrechte-Strategie praktisch keine Konflikte entstehen. Die einzige Situation, wo zu solchen Konflikten kommen kann, tritt in dem Fall auf, wenn die Editierrechte eines der Synchronisierungspartner aufgehoben wurden (siehe Abschnitt 4.2) und die entsprechenden Inhalte geändert wurden. Da die Entwicklung eines umfassenden Verfahrens zur Erkennung der textuellen Unterschiede und der Zusammenführung dieser Unterschiede den Rahmen dieser Arbeit überschreiten würde, wird ein einfaches Vorgehen für die Behandlung dieser Konflikte gewählt. Demnach wird der Benutzer über den Konflikt gewarnt, mit der Option, eine Sicherheitskopie der betroffenen Inhalte erstellen zu können. Danach werden diese Inhalte mit den neuen Daten überschrieben.

4.4.4 Gruppen-Management

Das zu entwickelnde Editiersystem soll das kollaborative Arbeiten in Gruppen unterstützen. Dementsprechend soll es für den Benutzer möglich sein, mehrere Gruppen zu erstellen und zu mehreren Gruppen zu gehören. Eine kollaborative Gruppe stellt damit einen virtuellen Arbeitsbereich dar, in dem nur die zugehörigen Personen agieren und miteinander kommunizieren können. Darüber hinaus sollen zu bearbeitende kollaborative Dokumente sich

unter anderem durch ihre Zugehörigkeit zu einer Gruppe auszeichnen. Dadurch wird garantiert, dass nur zugelassene Personen ein Dokument ansehen und bearbeiten können. Aus diesen Überlegungen wird ersichtlich, dass die Gruppenverwaltung, wie auch die im letzten Abschnitt beschriebene Synchronisierung der Dokumente, zu den wichtigsten Bestandteilen des kollaborativen Editiersystems zählt. Die damit verbundenen Aufgaben werden von dem Modul *Groups* übernommen. Dazu zählen:

- das Erstellen einer neuen Gruppe;
- das Hinzufügen von neuen Mitgliedern zu einer Gruppe;
- die Suche nach neuen Gruppen;
- das Betreten einer vorhandenen Gruppe;
- der Austritt aus der Gruppe;
- die Synchronisierung der Gruppendaten.

Darüber hinaus soll dieses Modul für andere System-Komponenten solche Funktionalität wie Speichern und Schließen der Gruppe anbieten.

Die Arbeit in einer Gruppe stellt mehrere Sicherheitsanforderungen an das kollaborative System. Dazu gehören z.B. Authentifizierung der Benutzer beim Betreten der Gruppe, Autorisierung der Gruppenmitglieder, Vertraulichkeit der Dokumente, Integrität der Daten etc. (nach [Eckert]). Um diesen Anforderungen gerecht zu werden, können solche Mechanismen wie Tickets, Zertifikate, Verschlüsselung oder Checksumme-Verfahren eingesetzt werden. Da die Entwicklung eines Sicherheitskonzeptes den Rahmen dieser Masterarbeit sprengen würde, wird hier nicht weiter darauf eingegangen. Stattdessen wird an dieser Stelle auf die Diplomarbeit von Horst Mund [Mund] verwiesen, in der das Thema der Berechtigungsstrukturen in kollaborativen Umgebungen ausführlich behandelt wurde.

Die Funktionalität des Moduls wird für andere System-Komponenten durch die Klasse *GroupManager* repräsentiert. Diese Klasse bildet damit die Schnittstelle zum *Groups*-Modul.

Die Gruppen werden im System für andere Komponenten, vor allem für die GUI-Schicht, durch zwei Objekte repräsentiert: *Group* und *Member*. Das Zusammenspiel der Objekte des *Groups*-Moduls ist im Klassendiagramm 4.21 zu sehen. Diese Objekte beinhalten alle Informationen über die Gruppe und ihre Mitglieder, die für das Agieren mit den Gruppen und das Darstellen der

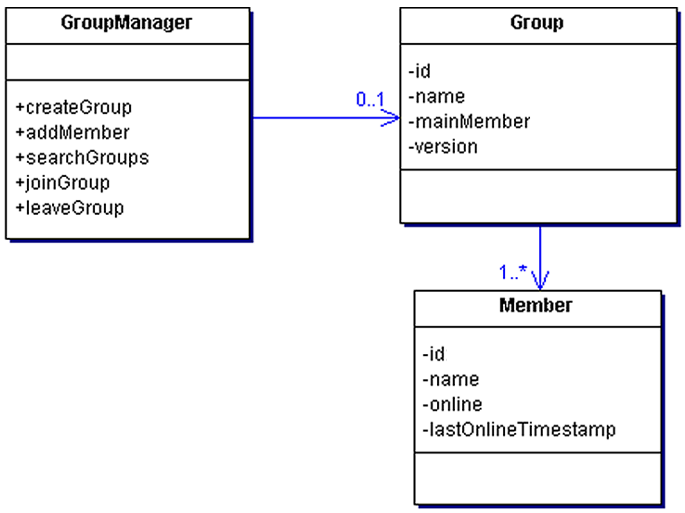

Abbildung 4.21: Klassendiagramm: Gruppen-Modell

Gruppe auf der Benutzer-Oberfläche notwendig sind. So besitzt ein Gruppen-Objekt die Attribute *Id* und *Name*. Des Weiteren enthält jede Gruppe die Information über ihre Mitglieder und die Id des Gruppenmitglieds, das diese Gruppe angelegt hat (*mainMember*). Damit wird erreicht, dass nur dieses und kein anderes Mitglied die Gruppe ändern kann, z.B. durch das Hinzufügen von weiteren Teilnehmern. Abgesehen von diesen Informationen besitzt das Gruppen-Objekt die Eigenschaft *version*, die für die Synchronisierung der Gruppendaten verwendet wird. Das Member enthält analog zum *Group*-Objekt ein Id- und ein Namen-Attribut. Des Weiteren besitzt das Member das Attribut *online*, das dem System-Benutzer signalisieren soll, ob der entsprechende Gruppenteilnehmer zurzeit online ist oder nicht. Diese Eigenschaft dient der Erhöhung der kollaborativen Awareness des Systems. Darüber hinaus existiert im Member die Eigenschaft *lastOnlineTimestamp*. Dieses Attribut dient zum Vermerken, wann das Member letztes Mal online war. Diese Information ist wichtig für die Überwachung der Gruppenmitglieder und für die Kontrolle darüber, wann Editierrechte aufgehoben werden können. Wenn das durch die Gruppe definierte Offline-Zeitlimit von einem Gruppenmitglied überschritten wird, wird der Initiator des Dokumentes oder der Autor der diesem Gruppenmitglied zugewiesenen Abschnitte darüber vom System in Kenntnis gesetzt. Darauf hat er die Möglichkeit, die Editierrechte aufzuheben.

Der *GroupManager* hält eine Referenz auf die aktuelle kollaborative Gruppe, die lokal oder durch die Suche erstellt oder aus den lokal vorhandenen Gruppen ausgewählt wurde.

Wie im Abschnitt 4.3.5 erläutert wurde, arbeitet das *Groups*-Modul eng mit der Kommunikations-Komponente zusammen. Jedes Mal, wenn eine neue Nachricht ankommt, wird das *Groups*-Modul darüber durch ein Event informiert. Darüber hinaus nimmt dieses Modul zusätzlich Dienste der Persistenz-Komponente in Anspruch, die die dauerhafte Speicherung der Gruppen übernimmt.

Erstellen einer neuen Gruppe

Das Erstellen einer neuen Gruppe kann von jedem Nutzer des Systems übernommen werden. Derselbe Teilnehmer trägt auch die Mitglieder in die Gruppe ein (siehe entsprechenden Anwendungsfall aus dem „Analyse“-Kapitel, Abschnitt 3.2.2). Eine Gruppe muss einen im System eindeutigen Namen besitzen. Aus diesem Grund erfolgt das Erstellen der Gruppe in folgenden Schritten: lokale Überprüfung, ob der Name der zu erzeugenden Gruppe schon vergeben ist; Überprüfung des Netzes auf die Existenz einer Gruppe mit dem gleichen Namen; Erzeugen des Gruppen-Objektes; Speichern des Objektes und Veröffentlichung der erstellten Gruppe im Netz. Dabei werden die Aufgaben, die mit der Netz-Kontrolle und -Veröffentlichung verbunden sind, an die Kommunikations-Komponente weitergeleitet. Der gesamte dynamische Ablauf ist in dem Diagramm 4.22 abgebildet.

Suche nach Gruppen

Abgesehen von dem Anlegen von neuen Gruppen steht dem Benutzer die Möglichkeit zur Verfügung, nach einer Gruppe im Netz zu suchen. Diese Funktionalität ist in erster Linie für die Teilnehmer von Bedeutung, die zu einer von ihrem anderen Team-Mitglied neu erstellten Gruppe beitreten wollen und ihre Daten deswegen benötigen. Da das System keine zentrale Komponente besitzt, soll eine passende Strategie dafür überlegt werden, wie diese Team-Mitglieder die Daten der Gruppe auf ihrem lokalen Editiersystem erhalten, um miteinander arbeiten zu können. Die Strategie besteht in diesem Editiersystem in einer dezentralen und asynchronen Suche nach Gruppen.

Bei der Suche nach Gruppen wird vorausgesetzt, dass die so genannten Interessenten von der Existenz ihrer Gruppe wissen und damit eine gezielte Suche im Netz durchführen können. Der *GroupManager* initiiert die Suche,

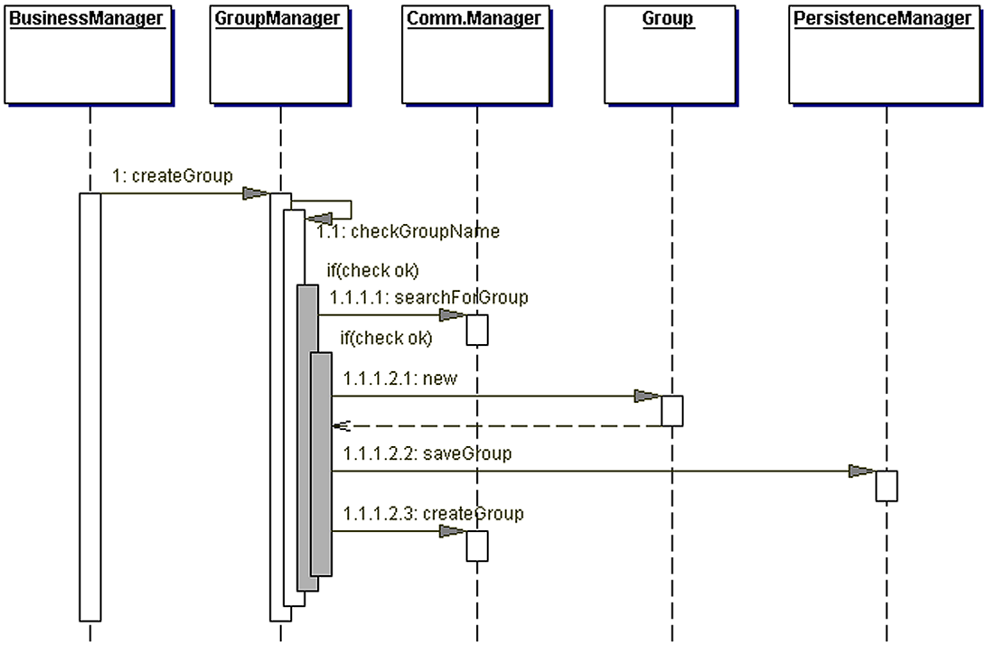

Abbildung 4.22: Sequenzdiagramm: Erzeugen einer neuen Gruppe

die dann von der Kommunikations-Komponente übernommen wird. Wenn eine Gruppe gefunden wurde (es ist mindestens ein Teilnehmer online, der diese Gruppe anbietet), wird das *Groups*-Modul darüber durch ein Event informiert. Darauf bekommt der suchende Benutzer eine entsprechende Benachrichtigung. Das dynamische Zusammenspiel der Module bei diesem Ablauf ist im Sequenzdiagramm 4.23 zu sehen.

Aus der Sicht des propagierenden Teilnehmers sieht der Ablauf folgendermaßen aus. Über das Ankommen der entsprechenden Anfrage wird der *GroupManager* durch die Kommunikations-Komponente informiert. Daraufhin wird überprüft, ob der anfragende Interessent zu der aktuellen Gruppe gehört. Wenn das der Fall ist, wird an den Interessenten eine entsprechende Nachricht übertragen. Im anderen Fall wird keine Nachricht an den Interessenten versendet. Dadurch wird garantiert, dass der suchende Benutzer nur die Gruppen findet, zu denen er auch gehören darf.

Bei der Suche werden nur die Gruppennamen übermittelt, keine anderen Informationen über die Gruppen. Die eigentlichen Daten der Gruppe werden erst bei deren expliziter Anforderung versendet, wenn der interessierende Teilnehmer der Gruppe beitreten will. Diese Maßnahme dient zur Verringerung des unnötigen Datentransfers.

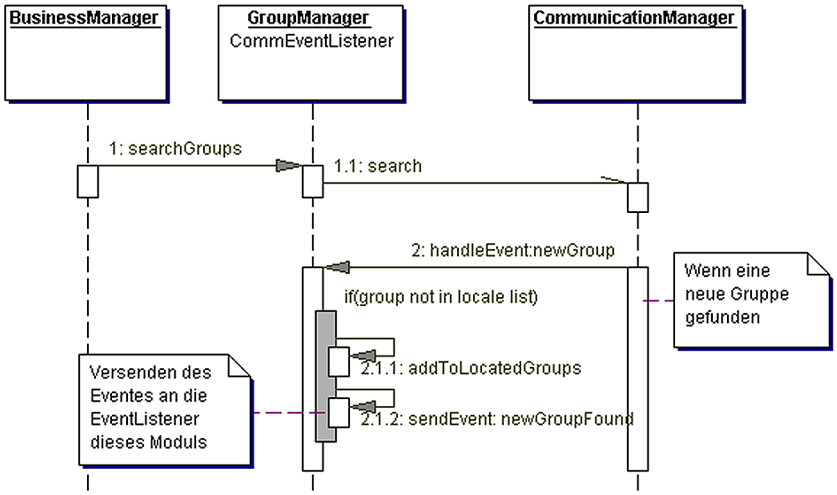

Abbildung 4.23: Sequenzdiagramm: Suche nach Gruppen

Betreten einer Gruppe

Dem Benutzer des Systems steht die Möglichkeit zur Verfügung, eine kollaborative Gruppe zu betreten. Durch diesen Schritt wird dem Benutzer erlaubt, ein Dokument, das zu dieser Gruppe gehört, auszuwählen, zu öffnen und zu bearbeiten. Dabei kann es sich entweder um eine lokal schon existierende oder eine durch die Suche gefundene Gruppe handeln. Dementsprechend unterscheiden sich die Abläufe voneinander, die in diesen beiden Fällen mit dem Betreten der Gruppe zusammenhängen.

Der gesamte Ablauf beim Betreten einer Remote-Gruppe aus der Sicht des Interessenten ist im Sequenzdiagramm 4.24 abgebildet.

Der erste Schritt beim Betreten einer im Netz gefundenen Gruppe liegt im Abruf der Daten dieser Gruppe auf das lokale Editiersystem. Dafür initiiert der *GroupManager* eine asynchrone Anforderungs-Aktion. Die Kontrolle des entsprechenden Ablaufs wird von der Kommunikations-Komponente übernommen. Der Teilnehmer, der die Gruppe besitzt, wird bei der angekommenen Anfrage darüber durch die Kommunikations-Komponente informiert und startet darauf die Übertragung der Gruppendaten. Unter diesen Daten werden nicht nur die Informationen der Gruppe selbst, sondern auch die eventuellen zu der Gruppe gehörenden Dokumente verstanden. Beim Erhalten der Daten auf der Seite des Interessenten wird der *GroupManager* darüber durch eine Nachricht in Kenntnis gesetzt. Darauf werden das Gruppen-Objekt erstellt und die Gruppe und ihre eventuellen Dokumente lokal gespeichert.

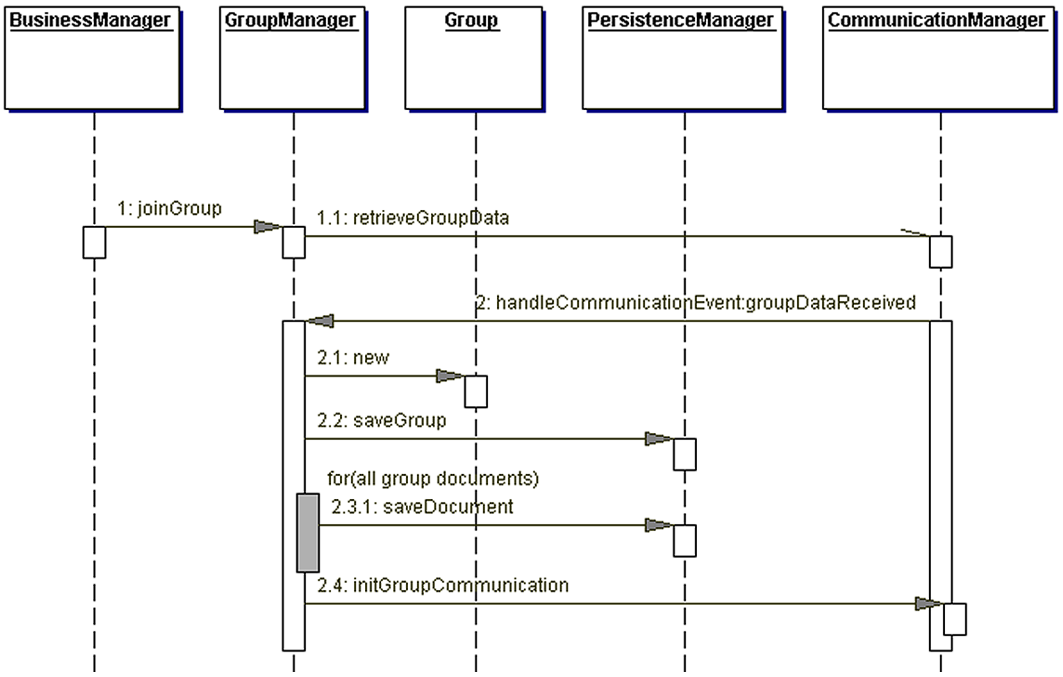

Abbildung 4.24: Sequenzdiagramm: Betreten der Gruppe

Beim Betreten einer lokal gespeicherten Gruppe werden die ausgewählte Gruppe geladen und ein entsprechendes Gruppen-Objekt erstellt.

Die letzten Schritte beim Betreten einer Gruppe sind sowohl für eine lokal existierende als auch für eine im Netz gefundene Gruppe identisch: Die Gruppen-Kommunikation wird initiiert und andere Gruppenmitglieder werden darüber in Kenntnis gesetzt, dass dieser Teilnehmer die Gruppe betreten hat. Diese Aufgaben werden von dem *GroupManager* an die Kommunikations-Komponente weitergeleitet (*initGroupCommunication* im Diagramm 4.24). Jetzt kann auch dieser Teilnehmer die Gruppe an ihre weiteren Mitglieder propagieren.

Aus der Sicht des propagierenden Teilnehmers besteht dieser Ablauf aus drei Schritten. Bei der ankommenden Anfrage wird zuerst überprüft, ob der Kommunikationspartner zur Gruppe gehört. Wenn das der Fall ist, wird die Übertragung der Gruppendaten gestartet. Außerdem wird die *lastOnlineTimestamp* des entsprechenden Members aktualisiert. Im anderen Fall wird an diesen Teilnehmer eine Nachricht versendet, die ihn über die Abweisung seiner Anfrage informiert.

Austritt aus der Gruppe

Das Austritt aus der Gruppe erfolgt nach folgendem Schema. Das betroffene Gruppenmitglied äußert den Austritt-Wunsch dem Gruppen-Initiator. Nach der positiven Bestätigung dieses Wunsches werden die Daten der Gruppe wie auch ihre Dokumente, die lokal bei diesem Benutzer existieren, gelöscht, da er kein Recht mehr hat, sie zu nutzen. Bei dem Gruppen-Initiator wird dieser Teilnehmer aus der Liste der Gruppenmitglieder entfernt. Die Editierrechte an den Dokumentabschnitten, die ihm zugewiesen waren, werden auf den Initiator übertragen, ähnlich dem Ablauf bei dem Aufheben der Editierrechte.

Synchronisierung der Gruppe

Unter der Synchronisierung der Gruppe wird die Aktualisierung ihrer Daten verstanden. Beim Betreten der Gruppe wird ihr aktueller Zustand (ihr Initiator, die Zusammensetzung der Gruppe usw.) bei einem der Online-Mitglieder abgefragt.

4.4.5 Kommunikations-Komponente

Die Kommunikations-Komponente spielt eine tragende Rolle bei Interaktionen zwischen den Teilnehmern des kollaborativen Editiersystems. Das Modul übernimmt folgende Aufgaben:

- Veröffentlichung einer neuen Gruppe im Netz;
- Beitreten zu einer Gruppe;
- Suche nach Gruppen im Netz;
- Austausch von Nachrichten zwischen den Teilnehmern;
- Versenden von Daten (Dokumenten und Gruppen).

Die Fassade zu diesem Modul stellt die Klasse *CommunicationManager* dar. Wie in dem Kapitel „Analyse“ erläutert wurde, wird in dieser Arbeit für die Kommunikation zwischen den einzelnen Benutzern des Systems das JXTA-Peer-to-Peer-Framework verwendet. Aus diesem Grund ist das Design des Kommunikations-Moduls entsprechend auf die Konzepte von JXTA ausgerichtet. Um trotzdem eine enge Abhängigkeit von dieser konkreten Technologie zu vermeiden, wird der *CommunicationManager* als ein Abstraktions-

schritt verwendet. Dieses Objekt leitet alle Aufrufe des Kommunikations-Moduls an ein weiteres Objekt weiter, das nach dem Strategie-Pattern [Gamma] durch eine abstrakte Schnittstelle *CommunicationIf* beschrieben wird. Dadurch kann die funktionale Klasse leicht ausgetauscht werden, wenn eine andere Umsetzung der Peer-to-Peer-Kommunikation erwünscht oder benötigt wird. In dieser Masterarbeit wird die Funktionalität des Moduls durch die Klasse *JxtaCommunication* umgesetzt. Die gesamte Klassenstruktur der Komponente und die entsprechenden statischen Beziehungen sind im Diagramm 4.25 dargestellt.

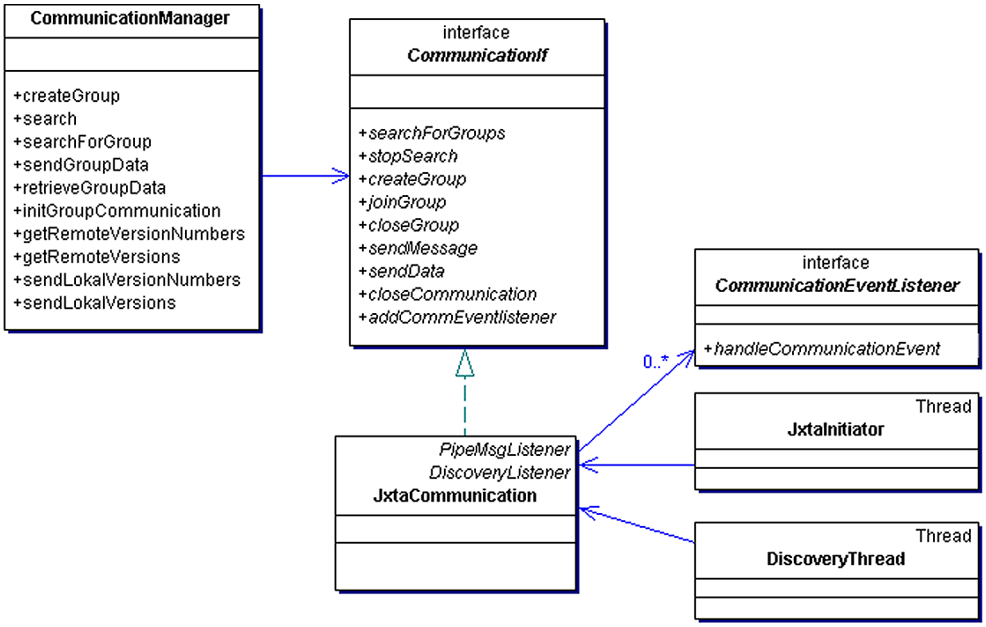

Abbildung 4.25: Klassendiagramm des Kommunikations-Moduls

Beim Starten des Editiersystems wird auch die Kommunikations-Komponente gestartet. Da dieser Vorgang einige Zeit in Anspruch nehmen kann, besonders wenn zu diesem Moment keine Verbindung zum Netz vorhanden ist, soll dieser Ablauf keine andere System-Funktionalität beeinträchtigen oder blockieren. Aus diesem Grund soll die Initiierung des Moduls in einem extra Thread erfolgen, dem *JxtaInitiator* (siehe Diagramm 4.25). Bei einem erfolgreichen Verbindungsaufbau werden andere System-Komponenten, die sich als *CommunicationEventListener* bei der Kommunikations-Komponente eingetragen haben, darüber durch eine Nachricht informiert.

Das Gleiche gilt auch für die Suche nach kollaborativen Gruppen: Da durch eine einmalige Suchanfrage nicht garantiert werden kann, dass die gesuchte

Gruppe gefunden wird, werden die Discovery-Nachrichten in regelmäßigen Abständen so lange versendet, bis eine Anweisung zum Stoppen der Suche kommt. Dazu kann es durch den expliziten Abbruch durch den Benutzer kommen oder dadurch, dass der Benutzer eine gefundene Gruppe zum Beitreten ausgesucht hat. Aus diesem Grund kann dieser Vorgang einige Zeit beanspruchen. Deswegen wird für diese Funktion ein zusätzlicher Thread gestartet, der *DiscoveryThread* (siehe Diagramm 4.25). Dadurch wird erreicht, dass die restliche Funktionalität des Systems von der Suche nicht beeinträchtigt wird.

Das JXTA-Framework verfügt über ein eigenes Gruppen-Paradigma, das nach einem hierarchischen Prinzip aufgebaut ist. Jeder JXTA-Teilnehmer, der so genannte *Peer*, gehört mindestens zu einer Peer-Gruppe. Am Anfang der Gruppen-Hierarchie steht die *NetPeerGroup*. Von dieser Gruppe können weitere Unter-Gruppen gebildet werden, die ihrerseits auch Bildung von Gruppen erlauben usw. Um die Möglichkeiten von JXTA maximal ausschöpfen zu können, wird in dem Kommunikations-Modul das Gruppen-Paradigma von JXTA eingesetzt. Demzufolge existiert zu jeder kollaborativen Gruppe aus dem *Groups*-Modul ein JXTA-Pendant in der Kommunikations-Komponente.

Zu den wichtigsten Diensten von JXTA gehören *PipeService* und *DiscoveryService*, die jede Peer-Gruppe besitzt. Der Pipe-Service dient zum Nachrichten-Austausch innerhalb einer Gruppe und der Discovery-Service zur Veröffentlichung von und der Suche nach Gruppen und Diensten im Netz. Da diese Services den größten Teil des oben beschriebenen Funktionsumfanges der Kommunikations-Komponente abdecken, werden sie in diesem Modul entsprechend verwendet. Für den Austausch von Nachrichten zwischen den Peers wird in JXTA ein asynchroner Mechanismus eingesetzt. Das gleiche gilt auch für den Discovery-Ablauf: Die Antworten von anderen JXTA-Instanzen kommen asynchron an. Dementsprechend wird die *JxtaCommunication* über die ankommenden Nachrichten durch bestimmte JXTA-Events benachrichtigt, worauf sie angemessen reagieren kann, z.B. durch das Versenden eigener Nachrichten an andere Module des Systems. Um auf die JXTA-Nachrichten hören zu können, implementiert diese Klasse zwei JXTA-Listener: *PipeMsgListener* und *DiscoveryListener* (siehe Klasse *JxtaCommunication* im Diagramm 4.25).

Ein weiterer JXTA-Dienst, den jede Peer-Gruppe besitzt und der in der Kommunikations-Komponente eingesetzt wird, ist *MembershipService*. Dieser Service wird für die Anfrage und das Akzeptieren von neuen Mitgliedschaften in einer Gruppe benötigt.

Standard-Mechanismen der JXTA-Kommunikation werden am Beispiel des Beitretens zu einer Peer-Gruppe erläutert. Diese Aktion wird initiiert, wenn der System-Benutzer die Absicht hat, eine kollaborative Gruppe zu betreten. Der Gesamt-Ablauf ist im Aktivitätsdiagramm 4.26 dargestellt.

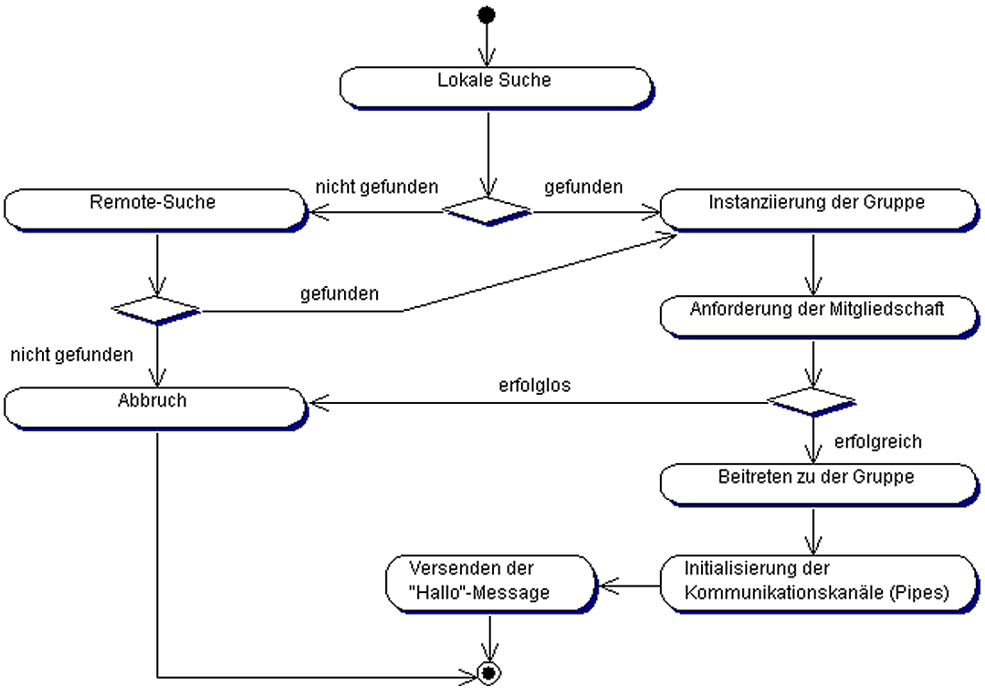

Abbildung 4.26: Aktivitätsdiagramm: Beitreten zu einer Peer-Gruppe

Der erste Schritt beim Beitreten zu einer Peer-Gruppe ist die lokale Suche nach dieser Gruppe (Aktivität *Lokale Suche* im Diagramm). Jede JXTA-Instanz führt einen Cache, in den schon bekannte Peer-Gruppen eingetragen werden. Wenn die Gruppe lokal nicht gefunden wurde, wird eine Remote-Anfrage gestartet (Aktivität *Remote-Suche*). Die beiden Suchen werden mit Hilfe des *DiscoveryService* durchgeführt. Wenn diese Suche auch erfolglos war, wird der Beitreten-Prozess beendet. Im anderen Fall wird eine neue Instanz der gefundenen Peer-Gruppe erzeugt (Aktivität *Instanziierung der Gruppe*). Im nächsten Schritt wird die Mitgliedschaft in der erstellten Gruppen-Instanz angefordert. Wenn die Anforderung erfolglos war, wird der Prozess beendet. Im anderen Fall wird der Peer-Gruppe beigetreten. Diese zwei Schritte erfolgen durch den *MembershipService* der Peer-Gruppe. Danach werden die Kommunikationskanäle der Gruppe (ihre In-und Out-Pipe) initialisiert. Das geschieht mit Hilfe des *PipeService* der Peer-Gruppe. Im

letzten Schritt wird über die Out-Pipe der Gruppe eine so genannte „Hallo"-Nachricht versendet, die andere, sich online befindende Gruppenmitglieder über den beigetretenen Benutzer informiert.

Das Versenden von Daten (Gruppendaten oder Dokumententeile) erfolgt wie auch das Verschicken von einfachen Nachrichten über den *PipeService* der entsprechenden Peer-Gruppe.

4.4.6 Das GUI-Design

Das GUI des Systems besteht aus zwei Modulen, wie es schon im Architektur-Überblick beschrieben wurde (Abschnitt 4.3): dem *Kollaborativen Editor* und einem Text-Editor nach Wahl des Benutzers.

Der *Kollaborative Editor* dient zur Präsentation der Arbeitsobjekte des kollaborativen Editiersystems (Dokumente und Gruppen) für den Benutzer und zur Interaktion des Benutzers mit dem System. Dementsprechend soll das GUI folgende Anforderungen erfüllen:

- Präsentation der Dokumentenstruktur und Interaktionen damit wie z.B. Anlegen neuer Kapitel, Vergabe der Editierrechte, Zufügen von Notizen an die einzelnen Abschnitte des Dokumentes usw.
- Präsentation der kollaborativen Informationen eines ausgewählten Dokumentabschnitts. Dazu gehören die Anzeige des Autors des Abschnitts, der aktuellen Editierberechtigung, des Datums der letzten Änderung usw.
- Präsentation der Annotationen zu einem ausgewählten Dokumentabschnitt und Interaktionen mit diesen, wie z.B. das Markieren der Annotationen als abgearbeitet
- Präsentation der kollaborativen Gruppe und Interaktionen mit ihr, wie z.B. Einfügen eines Mitglieds

Aus diesen Anforderungen können vier wesentliche Teile des Moduls abstrahiert werden, die sich voneinander durch ihre funktionalen Gebiete unterscheiden. Demnach kann die grafische Oberfläche in vier Bereiche unterteilt werden: „Dokumentbaum" für die Dokumentenstruktur, „Infobereich" für die kollaborativen Informationen des im Dokumentbaum ausgewählten Teils des Dokumentes, „Annotationen" für die Notizen und Anmerkungen des ausgewählten Teils des Dokumentes und „Gruppe" für die Gruppen-Informationen

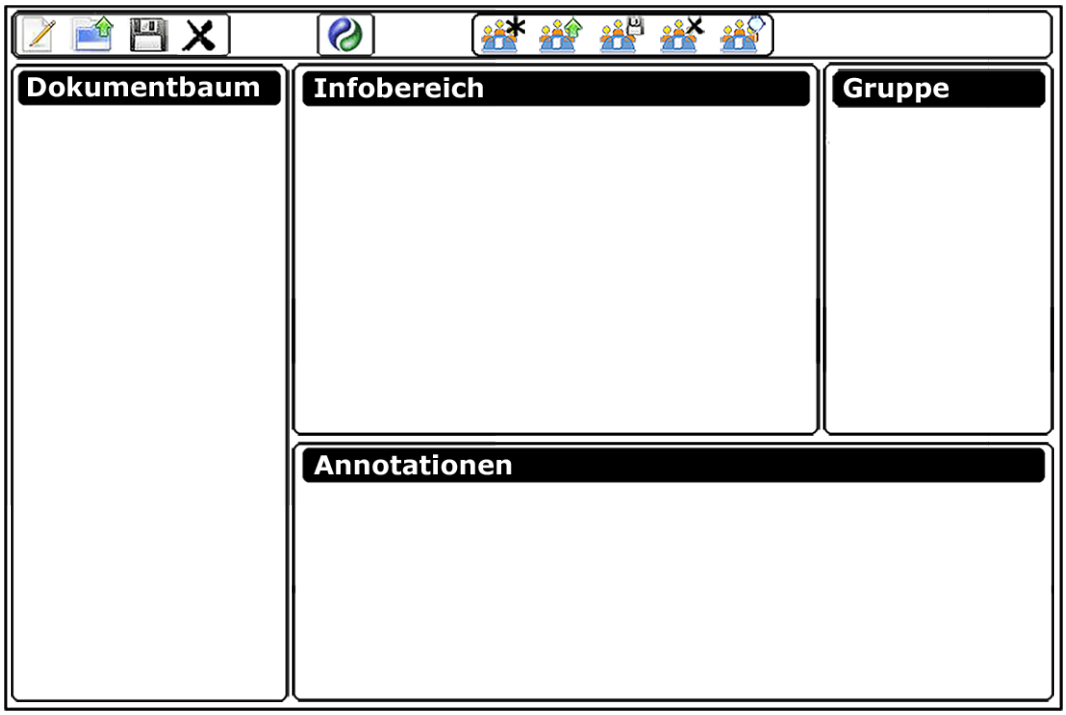

Abbildung 4.27: Die grafische Benutzeroberfläche des kollaborativen Editiersystems

(siehe Abbildung 4.27). In der oberen Leiste der Oberfläche sind Interaktionselemente zu sehen, die einige ausgewählte Funktionalität des Systems darstellen. Diese Elemente ermöglichen eine effiziente Arbeit mit kollaborativen Dokumenten und Gruppen. Wie zu sehen ist, sind diese Elemente in drei Bereiche unterteilt. Erste vier Interaktionselemente stellen Funktionen zur Verfügung bereit, die schnellen Zugriff auf die allgemeine Dokument-Funktionalität: ein neues Dokument anlegen, ein Dokument öffnen, speichern und schließen. Der zweite Bereich ermöglicht die Initiierung der Synchronisierung und der letzte Bereich steht für die allgemeine Gruppen-Funktionalität: Eine Gruppe erstellen, öffnen, speichern, schließen und suchen.

Die vier definierten Bereiche des *Kollaborativen Editors* werden entsprechend durch vier Klassen repräsentiert: *DocumentView*, *InfoView*, *AnnotationView* und *GroupView*. Gemäß den im Abschnitt 4.3.5 beschriebenen Abhängigkeiten zwischen dem GUI und der Business-Logik besitzen diese Klassen eine Referenz auf den *BusinessManager*. Außerdem implementieren sie das Interface *EventListener* und weisen einige weitere Gemeinsamkeiten auf, die mit dem Aufbau und dem Verhalten der grafischen Elemente zusammenhängen.

Aus diesem Grund wird eine abstrakte Oberklasse definiert, die gleiche Eigenschaften dieser Klassen vereint: die Klasse *AbstractView*. Die Unterschiede im Design dieser Bereiche sind mit der Art der Daten, die sie präsentieren sollen, und mit der Art dieser Präsentation verbunden. Die GUI-Klassen referenzieren zur Darstellung von Daten die entsprechenden Objekte der Business-Logik: *DocumentView*, *AnnotationView* und *InfoView* beziehen sich auf den *DocumentManager* und *GroupView* hält eine Referenz auf das *GroupManager*-Objekt. Im Klassendiagramm 4.28 sind einzelne Bestandteile des GUI zu sehen.

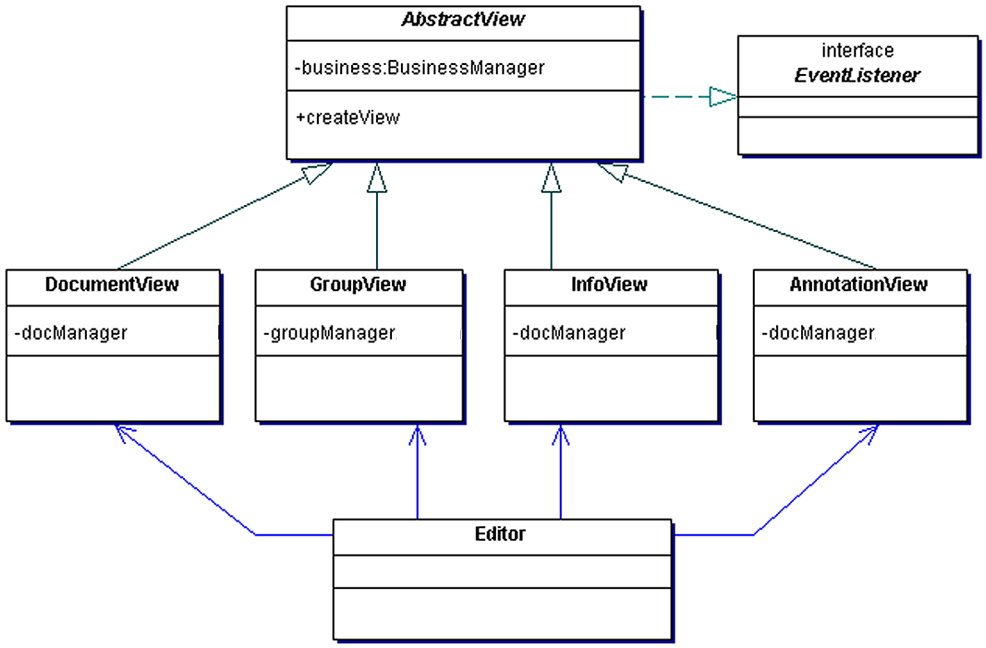

Abbildung 4.28: GUI: Klassendiagramm

4.5 Realisierung

Im Rahmen dieser Masterarbeit wurde ein Prototyp des kollaborativen Editiersystems implementiert. In diesem Prototyp wurde nicht das komplette ausgearbeitete Konzept des Systems umgesetzt, sondern seine grundlegende Funktionen, die die Tragfähigkeit des Systems zeigen sollten, z.B. Erstellen von Gruppen und Dokumenten, Synchronisierung der Dokumente, Beitreten zu einer Remote-Gruppe usw. In diesem Abschnitt werden interessante

Aspekte dieser Realisierung präsentiert und es wird am Beispiel eines Anwendungsfalls und anhand von Screenshots der grafischen Oberfläche der Anwendung die Funktionsweise des Systems gezeigt.

4.5.1 Allgemeines

Die Anwendung wurde in der Programmiersprache Java, Version 1.6, implementiert. Für die Entwicklung wurde die Eclipse-Entwicklungsumgebung [Eclipse] in der Version 3.3.0 verwendet.

Für das XML-Objekt des Dokument-Modells wurden Objekte von JDOM [JDOM], einem Java-API zur Arbeit mit XML-Dokumenten, eingesetzt.

Für die Navigation und den Zugriff auf die Elemente der XML-Objekte wurde die Sprache XPath [XPath] eingesetzt. Dafür wurde ihre Implementierung von JDOM verwendet.

Für die Implementierung der graphischen Oberfläche wurde das SWT(Standard Widget Toolkit)-Framework [SWT] von Eclipse, Version 3.3, verwendet.

Als Peer-to-Peer-Kommunikationsplattform wurde die Java-Referenz-Implementierung von JXTA, das JXSE Projekt [JXSE] von Sun Microsystems, in der Version 2.5 benutzt.

Als ein Beispiel-Editor für LaTeX-Dokumente wurde die Shareware-Version des Editors WinEdt [WinEdt] in der Version 5.4 genommen.

4.5.2 Persistenz

Wie schon im Entwurf-Teil erläutert wurde, wurde in dieser Arbeit ein Dateibasierter Ansatz für die Datenhaltung gewählt. Dafür wurde eine spezielle Verzeichnisstruktur ausgearbeitet, die in der Abbildung 4.29 zu sehen ist. Die Struktur besteht aus zwei Verzeichnissen: für Gruppen und Dokumente. In der Abbildung sind diese Verzeichnisse als *groups* und *coll-documents* gekennzeichnet. Wie auch kollaborative Dokumente werden Gruppen in einem XML-Format gespeichert, das alle Informationen der korrespondierenden *Group*-Objekte enthält, abgesehen von der dynamischen „online“-Eigenschaft der Members. Die Daten aller Gruppen des System-Benutzers werden in einer Datei im Verzeichnis *groups* gehalten.

Das Verzeichnis *coll-documents* enthält Dokumenten-Daten, aufgeteilt nach den Gruppen, zu denen diese Dokumente gehören. Dementsprechend enthält

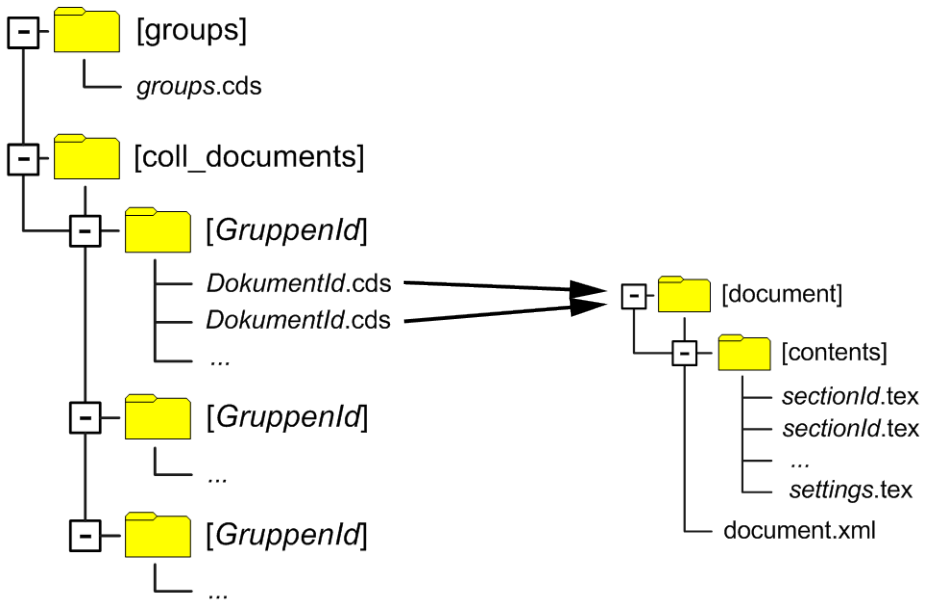

Abbildung 4.29: Verzeichnisstruktur für kollaborative Dokumente

Dokumenten-Verzeichnis so viele Unterverzeichnisse wie kollaborative Gruppen, in denen der Benutzer mitarbeitet. Diese Verzeichnisse sind nach der Id der jeweiligen Gruppe genannt, um die Suche nach dazugehörigen Dokumenten zu erleichtern. In der Abbildung sind sie als Verzeichnisse mit dem Namen *GruppenId* zu sehen.

Jedes Dokument wird als ein Verzeichnis mit einer bestimmten, für alle Dokumente des Editiersystems gleichen Struktur gespeichert (siehe Verzeichnis *document* in der Abbildung). Nach diesem Ansatz besteht das Dokument aus einer XML-Datei *document.xml*, die das kollaborative Dokument abbildet, und einem Verzeichnis mit den fachlichen Inhalten des Dokumentes (in dieser Arbeit LaTeX-Dateien), das in der Abbildung als *contents* zu sehen ist. Jedes Teildokument trägt seine Id als Namen für eine effiziente Suche nach bestimmten Dokumentabschnitten.

Bei jeder Datenhaltung muss garantiert werden, dass Daten von unkontrollierten Benutzer-Zugriffen geschützt bleiben. Bei einem Datei-basierten Persistenz-Ansatz ist für es den Benutzer in der Regel leichter, an die Daten zu kommen, als bei der Datenhaltung in einer Datenbank. Da die Dokumente in diesem Editiersystem ein textuelles Format haben, mussten Maßnahmen vorgenommen werden, die den Schutz der Daten erhöhen. Sowohl Dokumente als auch Gruppendaten werden aus diesem Grund in einer komprimierten

Form gehalten und können nach Bedarf verschlüsselt werden. In der Abbildung 4.29 sind sie als Dateien mit der Erweiterung *.cds* zu sehen. Beim Laden eines Dokumentes oder der Gruppen werden die entsprechenden Daten in ein temporäres Verzeichnis ausgepackt. Nach dem Schließen des Dokumentes und / oder des Programms wird der Inhalt dieses Verzeichnisses gelöscht. Für die Komprimierung wurde die Standard-Bibliothek von Java für die Zip-Komprimierung verwendet. Die entsprechende Funktionalität wurde in einer Util-Klasse *Zip* implementiert, die Methoden sowohl für das Packen als auch für das Entpacken von Dateien zur Verfügung stellt.

4.5.3 Kommunikations-Komponente

Die spezifischen Eigenschaften von JXTA, so z.B. die automatische Zugehörigkeit einer Peer-Gruppe zu der *NetPeerGroup* und das lokale Caching von Informationen über Peer-Gruppen (mehr dazu siehe in [Michael, JxtaProgGuide]), machen es möglich, eine Peer-Gruppe auch dann zu finden, wenn kein Peer aktuell zu ihr gehört. Da aber die Suche nach kollaborativen Gruppen im Editiersystem dazu dient, die Daten der neu gefundenen Gruppe lokal zu erhalten (siehe Abschnitt 4.4.4), muss garantiert werden, dass mindestens ein Benutzer existiert, der die Gruppe betreten hat und die Daten dieser Gruppe für andere anbietet. Das ist die Voraussetzung für ein korrektes Verhalten des Systems. Aus diesem Grund wurden ein spezielles Gruppen-Schema für das Editiersystem und eine Strategie für die Suche nach Gruppen innerhalb des Systems ausgearbeitet:

- Für das gesamte Editiersystem existiert eine Hauptgruppe: die *CollaborativeGroup*. Die Peer-Gruppen, die tatsächliche kollaborative Gruppen abbilden, existieren als Untergruppen dieser Gruppe. Jeder Benutzer, der die Anwendung startet, gehört automatisch zu der *CollaborativeGroup*.

- Durch die Suche werden nur aktive Gruppen gefunden. Unter einer aktiven Gruppe wird dabei eine Gruppe verstanden, bei der mindestens ein Mitglied zur Zeit der Suche online ist.

- Die Suche nach kollaborativen Gruppen soll innerhalb des Raumes der *CollaborativeGroup* stattfinden. Sie erfolgt dabei durch Versenden einer bestimmten Nachricht über die Out-Pipe dieser Haupt-Gruppe. Als Reaktion auf diese Discovery-Nachricht versendet der andere System-Teilnehmer eine Antwort-Message, in der er den Namen seiner aktuellen Gruppe übermittelt, wenn er dieser beigetreten ist. Damit bekommt

der Suchende den Überblick über die zurzeit aktiven kollaborativen Gruppen.

Die Kommunikation zwischen den System-Teilnehmern erfolgt in Form von JXTA-Messages. Eine JXTA-Message ist die Basis-Einheit für den Austausch von Daten zwischen JXTA-Peers [JxtaProgGuide]. Sie besteht im Allgemeinen aus einer geordneten Sequenz von mehreren Message-Elementen. Jedes Element stellt ein Name-Werte-Paar dar. Innerhalb des Editiersystems werden für die Kommunikation *Propagate*-Pipes eingesetzt, die das Versenden von Messages an mehrere Teilnehmer gleichzeitig ermöglichen. Dadurch wird garantiert, dass alle Gruppenmitglieder solche Nachrichten wie z.B. die Bekanntmachung des Beitretens eines neuen Mitglieds zu der Gruppe erhalten. Es existiert aber auch solche Kommunikation innerhalb des Systems, die zwischen nur zwei Teilnehmern stattfinden soll. Z.B. soll bei der Anfrage der Gruppendaten nur ein Teilnehmer antworten und diese Daten senden. Das Gleiche gilt auch für die Synchronisierung der Dokumente. Um den überflüssigen Datentransfer zu verhindern, wurde für Messages, die innerhalb des Editiersystems ausgetauscht werden, ein spezielles Format ausgearbeitet.

Demnach besteht eine Nachricht aus folgendem Inhalt:

- Schlüssel der Nachricht (*Key*);
- Autor der Nachricht (*Author*);
- Empfänger der Nachricht (*Receiver*);
- Daten der Nachricht (*Data*);
- Datenlänge (*Length*);
- Paket-Nummer (*PackageNumber*).

Key gibt eindeutig an, um welche Art von Nachricht es sich handelt, z.B. der Schlüssel *GetGroup* identifiziert die Nachricht als Such-Anfrage, *MyGroup* als Antwort auf diese Anfrage usw. Wenn in einer Nachricht ein bestimmter *Receiver* vermerkt ist, wird diese Nachricht nur von diesem Peer beantwortet. Wenn das Feld leer bleibt, werden alle Teilnehmer antworten, die diese Nachricht bekommen haben. *Author* identifiziert den Kommunikationspartner und gibt an, an wen die Antwort versendet werden soll. Die letzten Elemente *Length* und *PackageNumber* sind für die Übertragung von einer großen Menge Daten nützlich: Eine JXTA-Message ist auf 64 KB begrenzt

[JxtaProgGuide], deswegen müssen größere Datenmengen in Form von mehreren Messages versendet werden. Um die einzelnen Teile auf der Empfänger-Seite wieder zusammenführen zu können, werden diese beiden Elemente verwendet.

4.5.4 Datenübertragung

Der Austausch von größeren Daten zwischen den System-Nutzern findet in komprimierter Form statt, um das Übertragungsvolumen zu verringern. Zu solchen Daten gehören Daten einer Gruppe oder eines Dokumentes, Versionsnummern der Dokumente und Id's der für die Aktualisierung des Dokumentes benötigten Dokumentabschnitte (siehe Abschnitt 4.4.3). Die Daten werden mit Hilfe der im vorherigen Abschnitt beschriebenen Klasse *Zip* komprimiert.

Die Gruppendaten werden in Form einer XML-Datei an den Kommunikationspartner übertragen. Das XML-Format entspricht dem Format, in dem die Gruppe lokal gespeichert wird.

Der Austausch der Dokumente zwischen zwei Synchronisierungsmodulen erfolgt in Form einer Datei, in der die entsprechenden Daten als weitere Dateien verpackt werden. Die Verzeichnis-Struktur dieser Daten ist mit der Struktur identisch, die lokal für die Persistenzierung der Dokumente verwendet wird (siehe vorherigen Abschnitt). Es werden gewünschte fachliche Inhalte und eine XML-Datei übertragen, die dem Schema des kollaborativen Dokumentes entspricht (siehe Abschnitt 4.1.5), aber nur die angeforderten Teile des Dokumentes enthält. Durch diese Vorgehensweise wird garantiert, dass das anfordernde Synchronisierungsmodul die ganzen benötigten Daten erhält.

4.5.5 Beispielablauf

Die Funktionsweise der entwickelten Anwendung „Kollaborativer Editor“ wird am Beispiel eines Ablaufs vorgestellt, der sich aus mehreren, im Abschnitt 3.2.2 beschriebenen Anwendungsfällen zusammensetzt. Dieser Ablauf stellt einige typische Schritte dar, die bei der kollaborativen Erstellung von Dokumenten durchgeführt werden, und zeigt die Anwendung aus der Perspektive von zwei Gruppenmitgliedern: *Jan* und *Lisa*, die aus dem Beispielszenario (Abschnitt 3.2.1) bekannt sind.

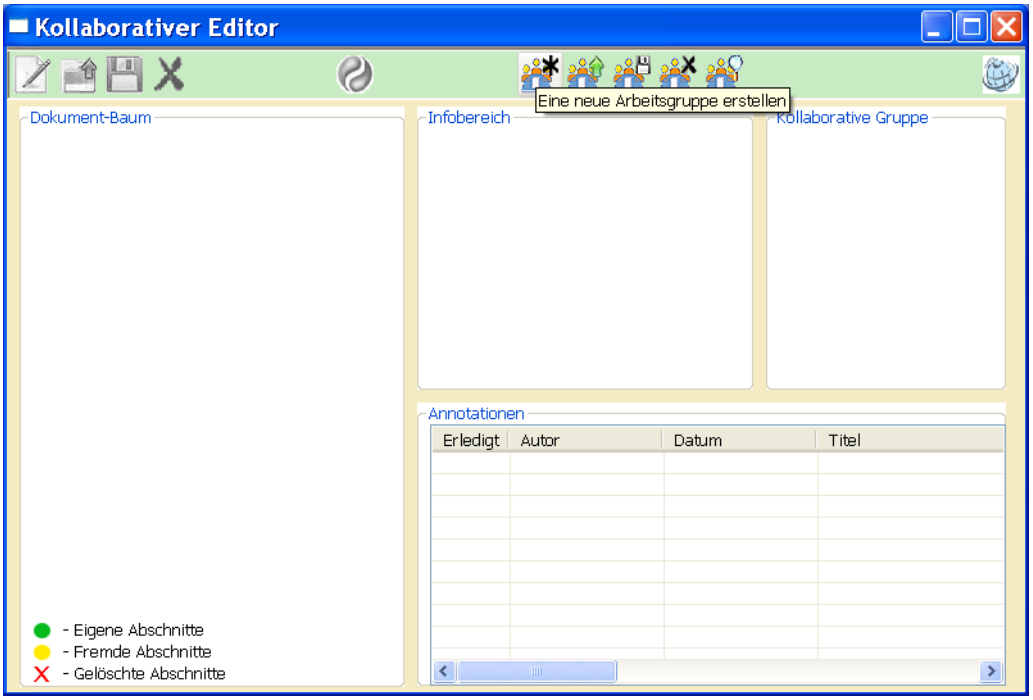

Abbildung 4.30: Grafische Oberfläche der Anwendung

Der Beispielablauf wird anhand einer Folge von Screenshots der Anwendung vorgestellt.

Die erste Abbildung (4.30) zeigt die grafische Oberfläche nach dem Start des Programms. Das Symbol in der rechten oberen Ecke zeigt an, ob der Benutzer zurzeit online ist. Die Buttons für das Arbeiten mit dem Dokument in der oberen Leiste sind deaktiviert, da der Benutzer zurzeit keiner Gruppe beigetreten ist. Um mit einem Dokument arbeiten zu können, muss der Benutzer zuerst eine kollaborative Gruppe entweder aus der lokalen Liste auswählen, durch die Suche im Netz finden oder neu erstellen.

Abbildung 4.31: Erstellen einer neuen Gruppe

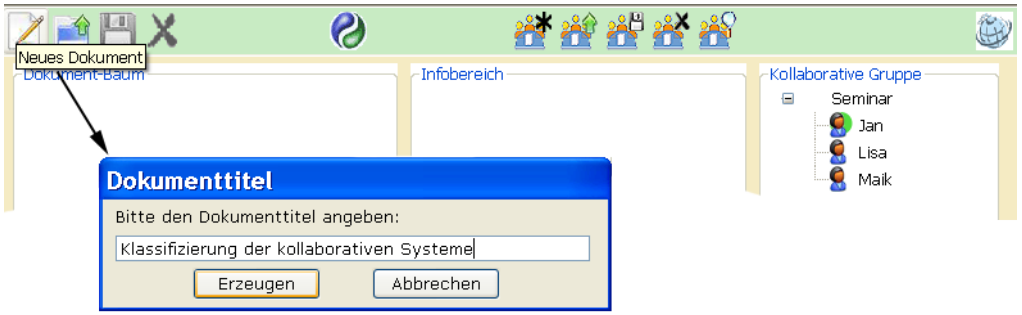

Abbildung 4.32: Anlegen eines neuen Dokumentes

In diesem Fall ist der Benutzer der Anwendung Jan. Durch ihn wird eine neue Gruppe erzeugt. Dieser Schritt ist in der Abbildung 4.31 zu sehen. Beim Anlegen der Gruppe werden unter anderem ihre Mitglieder eingetragen. Bei diesem Ablauf sind das Jan, Lisa und Maik aus dem Beispielszenario. Sie haben vor, eine Seminar-Ausarbeitung zum Thema „Klassifizierung der kollaborativen Systeme“ zusammen zu schreiben, deswegen wird der Gruppe der Name „Seminar“ vergeben.

Nachdem die Gruppe angelegt wurde, erscheinen ihr Name und ihre Zusammensetzung im Bereich *Kollaborative Gruppe* der grafischen Oberfläche (siehe Abbildung 4.32). Da Jan die Gruppe erst erstellt hat und andere Gruppenmitglieder ihre Daten noch nicht haben, ist nur er als Online-Mitglied durch ein entsprechendes Icon markiert.

Im nächsten Schritt erzeugt Jan ein neues Dokument für diese Gruppe mit dem Namen „Klassifizierung der kollaborativen Systeme“ (siehe Abbildung 4.32).

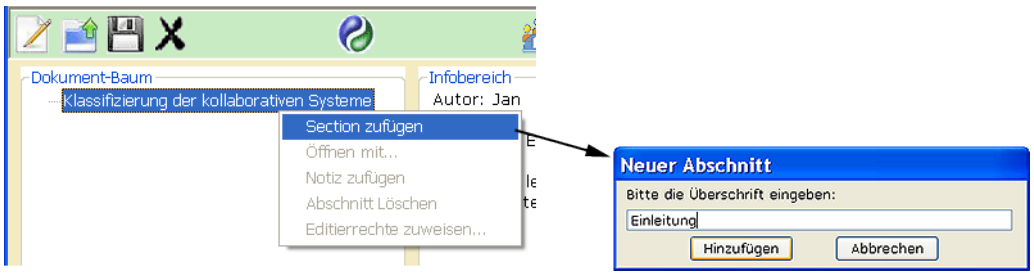

Abbildung 4.33: Anlegen der Struktur des Dokumentes 1

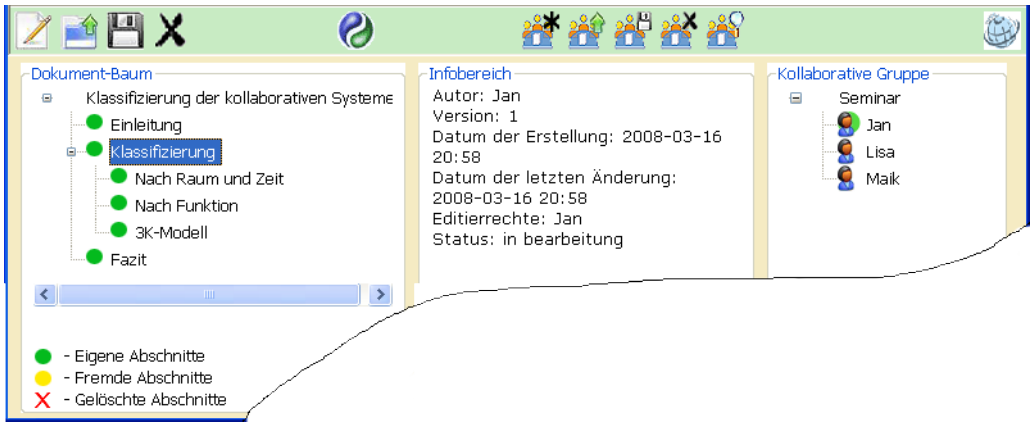

Abbildung 4.34: Anlegen der Struktur des Dokumentes 2

Nach der Erzeugung des Dokumentes wird seine vorläufige Struktur angelegt. Das geschieht durch die mehrfache Ausführung der Aktion *Section zufügen* und der Vergabe einer Überschrift für den neuen Abschnitt (Abbildung 4.33). In diesem Fall besteht das Dokument vorläufig aus drei Teilen: „Einleitung“, „Klassifizierung“ und „Fazit“. Der Abschnitt „Klassifizierung“ enthält weitere drei Unterabschnitte „Nach Raum und Zeit“, „Nach Funktion“ und „3K-Modell“. In der Abbildung 4.34 ist die fertige Struktur des Dokumentes in Baumform zu sehen. Durch die Icons an den Knoten des Baums wird dem Benutzer gezeigt, auf welche Dokumentabschnitte er die Editierrechte hat und auf welche nicht. Unterschiedliche Editierrechte werden dabei durch verschiedenfarbige Icons dargestellt. Für einen gelöschten Abschnitt existiert

ein extra Icon. Nach der Erstellung des Dokumentes gehören alle Editierrechte Jan, deswegen werden alle Dokumentabschnitte als eigene gekennzeichnet (siehe Abbildung 4.34).

Beim Auswählen eines Abschnitts im Dokumentbaum werden seine kollaborativen Informationen im Infobereich angezeigt, wie es in der Abbildung 4.34 zu sehen ist. Unter anderem gehören dazu die Angaben über den Autor dieses Abschnitts, über den Status der Bearbeitung und über die aktuellen Editierrechte.

Im nächsten Schritt werden Editierrechte aufgeteilt. Das kann durch die Markierung des entsprechenden Abschnitts, die Wahl der Aktion *Editierrechte zuweisen...* und die Auswahl des Gruppenmitglieds, dem die Editierrechte zugewiesen werden müssen, durchgeführt werden. Lisa bekommt die Editierrechte für den Abschnitt „Klassifizierung“ (siehe Abbildung 4.35). Entsprechend der

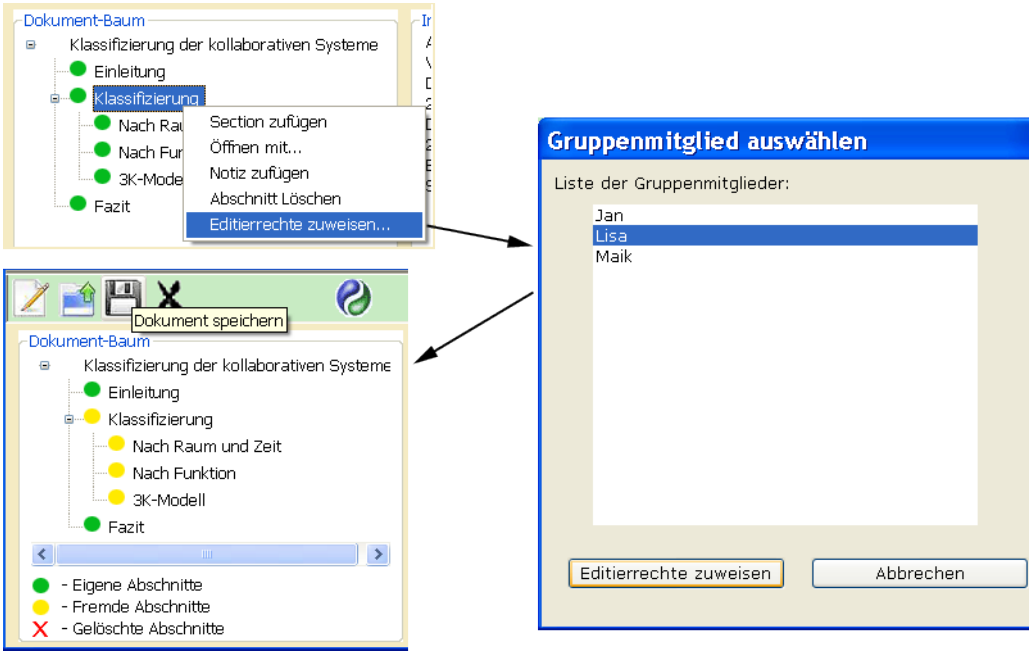

Abbildung 4.35: Zuweisen der Editierrechte

Rechtevergabe-Strategie (Abschnitt 4.2) werden an Lisa auch die Editierrechte auf die Unterabschnitte dieses Dokumententeils automatisch übertragen. Nach der Zuweisung der Rechte werden entsprechende Knoten im Dokumentbaum durch die Icons in einer anderen Farbe dargestellt (siehe Abbildung 4.35).

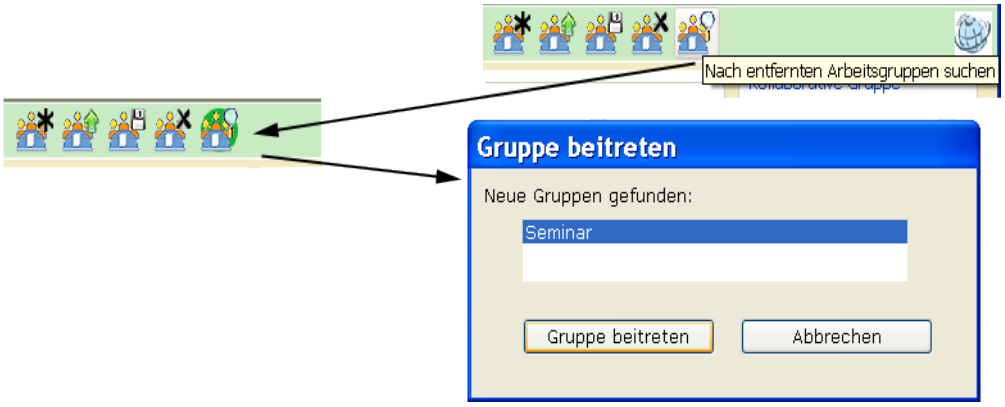

Abbildung 4.36: Suche nach der Gruppe

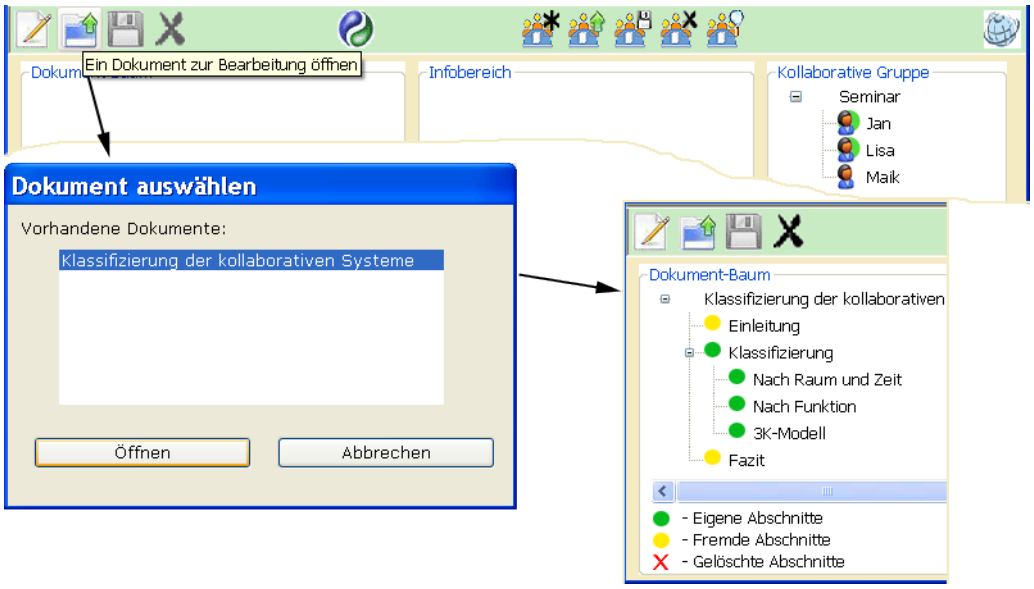

Abbildung 4.37: Öffnen des Dokumentes

Der nächste Teil des Ablaufs wird aus der Perspektive eines anderen Gruppenmitglieds, Lisa, dargestellt. Zuerst muss Lisa die Daten der von Jan erstellten Gruppe abrufen. Dafür startet sie die Suche nach entfernten Arbeitsgruppen (siehe Abbildung 4.36).

Durch das Klicken des entsprechenden Symbols auf der Oberfläche wird die Suche aktiviert und nachdem eine oder mehrere Gruppen gefunden wurden, erscheint ein Fenster mit ihren Namen. In diesem Fall ist es die Gruppe „Seminar“, die im Netz gefunden wurde.

Durch das Markieren der gesuchten Gruppe und dem Betätigen des Buttons *Gruppe beitreten* wird der Vorgang initiiert, im dessen Verlauf die Daten dieser Gruppe und ihre Dokumente auf das lokales System übertragen werden und der Benutzer der Gruppe beitritt. Als Ergebnis dieser Aktion erscheinen im Bereich *Kollaborative Gruppe* die entsprechenden Gruppendaten. Dabei werden als Online-Mitglieder der aktuelle Benutzer selbst und die Gruppenmitglieder markiert, die auf seine *Hallo*-Nachricht geantwortet haben. In diesem Fall sind das Lisa und Jan, der ihr die Gruppendaten zur Verfügung gestellt hat (siehe Abbildung 4.37).

Jetzt kann Lisa das gemeinsame Dokument öffnen und mit ihm arbeiten. Das Öffnen des Dokumentes geschieht durch das Betätigen des Buttons *Dokument öffnen* und die Auswahl des benötigten Dokumentes (siehe Abbildung 4.37). Nach der Bestätigung der Auswahl erscheint im Bereich *Dokument-Baum* der grafischen Oberfläche die Struktur des ausgewählten Dokumentes. Dabei werden die Abschnitte des Dokumentes, auf die Lisa die Editierrechte hat, durch das entsprechende Icon als eigene ausgezeichnet.

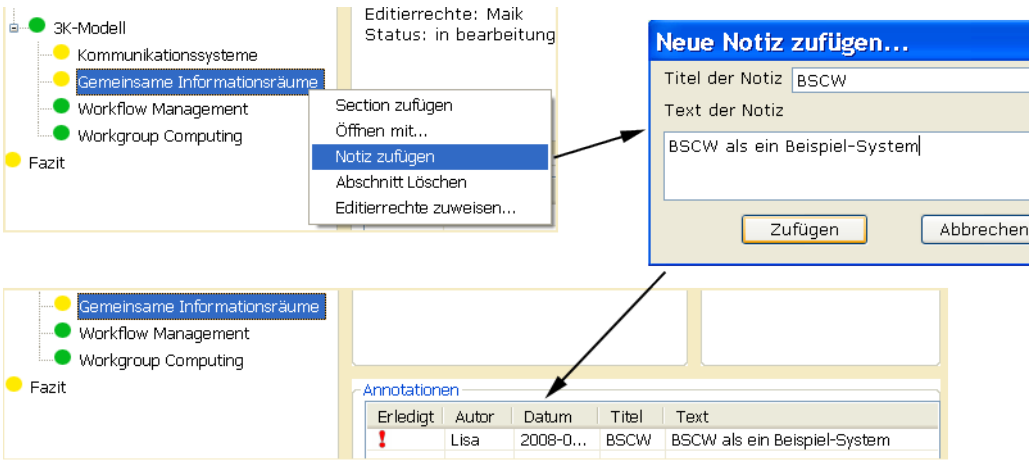

Abbildung 4.38: Zufügen einer Annotation

Lisa erstellt unter dem ihr zugewiesenen Abschnitt „3K-Modell“ weitere vier Unterabschnitte: „Kommunikationssysteme“, „Gemeinsame Informationsräume“, „Workflow Management“ und „Workgroup Computing“. Die Editierrechte an zwei von diesen Abschnitten („Kommunikationssysteme“ und „Gemeinsame Informationsräume“) weist sie einem weiteren Gruppenmitglied zu, Maik. Außerdem fügt sie dem Abschnitt „Gemeinsame Informationsräume“ eine Notiz zu. Das geschieht durch das Markieren des entsprechenden Abschnitts und der Auswahl der Aktion *Notiz zufügen*. Darauf erscheint ein

Fenster, wo der Name und der Text der Notiz eingegeben werden können. Nach der Bestätigung erscheint die neue Notiz im Bereich *Annotationen* der grafischen Oberfläche der Anwendung. Der ganze Ablauf ist in der Abbildung 4.38 zu sehen.

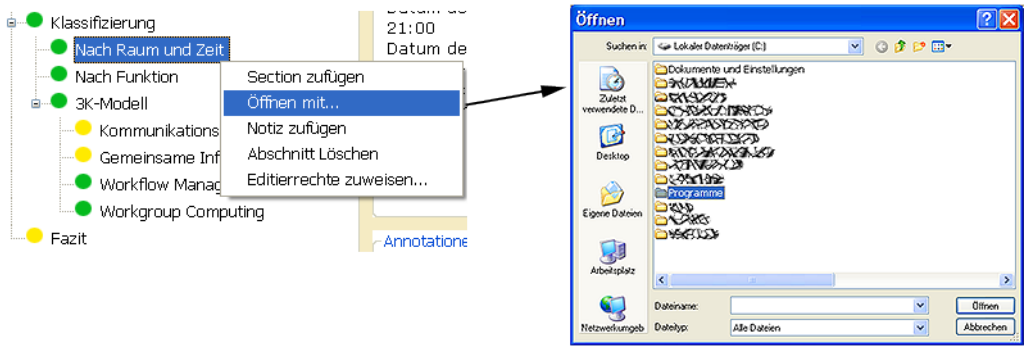

Abbildung 4.39: Öffnen eines Abschnitts zum Editieren 1

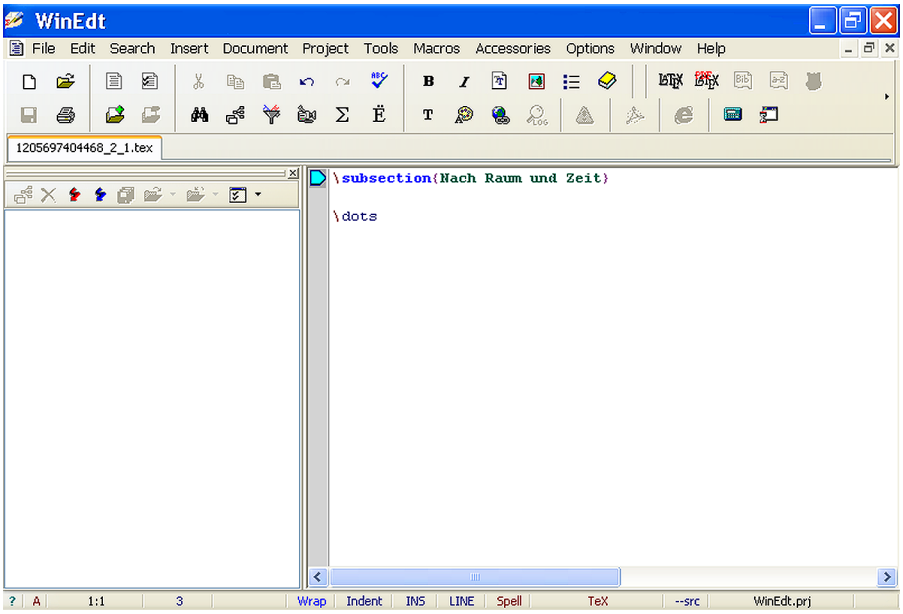

Abbildung 4.40: Öffnen eines Abschnitts zum Editieren 2

Der nächste Schritt beinhaltet das Öffnen eines Dokumentabschnitts in einem Editor zur Bearbeitung. Lisa hat vor, den Abschnitt „Nach Raum und Zeit“ zu bearbeiten. Dafür wird der entsprechende Abschnitt markiert und die Aktion *Öffnen mit...* ausgewählt. Darauf erscheint ein Fenster, wo ein Textbearbeitungsprogramm zum Editieren ausgewählt werden kann (siehe

Abbildung 4.39). Nach der Bestätigung der Auswahl werden das ausgewählte Programm gestartet und der zu bearbeitende Abschnitt automatisch zum Editieren geöffnet (siehe Abbildung 4.40). Bei diesem Beispiel-Ablauf wird die Shareware-Version des Programms WinEdt zum Schreiben am Dokument eingesetzt.

Der letzte Teil des Ablaufs wird aus der Perspektive von Jan dargestellt und beinhaltet die Synchronisierung des gemeinsamen Dokumentes.

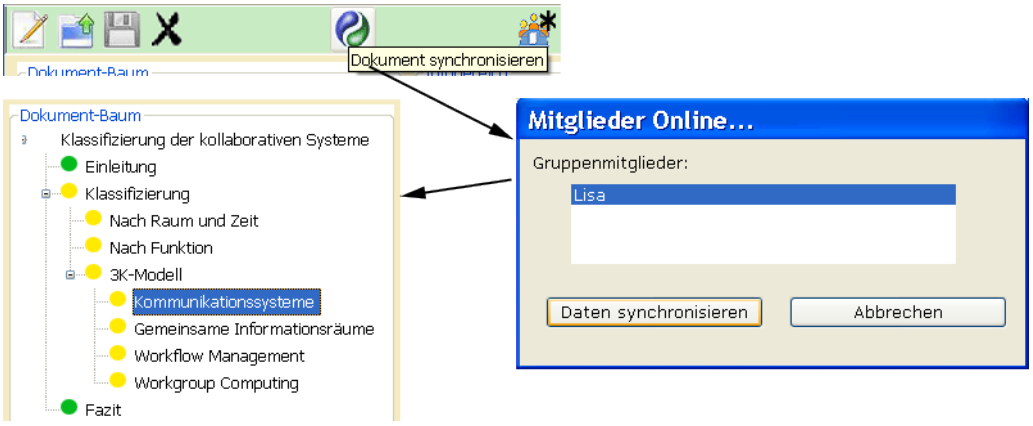

Abbildung 4.41: Synchronisierung des Dokumentes

Jan will seine Dokument-Version aktualisieren. Dafür betätigt er den Button *Dokument synchronisieren* in oberer Leiste der grafischen Oberfläche. Darauf erscheint ein Fenster, wo alle Gruppenmitglieder aufgelistet sind, die zurzeit online sind (siehe Abbildung 4.41). In diesem Fall ist es Lisa. Nach der Auswahl des Kommunikationspartners und der Bestätigung dieser Auswahl startet die Synchronisierung. Nach dem Beenden dieses Vorgangs erscheint bei Jan die aktuelle Version des Dokumentes, einschließlich der Erweiterungen von Lisa (siehe Abbildung 4.41).

4.6 Evaluation

Das im Rahmen dieser Masterarbeit entwickelte Konzept sollte ermöglichen, in einer Gruppe kollaborativ und dezentral Dokumente zu bearbeiten. Die im Kapitel „Analyse“ gestellten Anforderungen wurden folgendermaßen erfüllt:

- **Gruppen-Verwaltung**
  Das in dieser Arbeit entworfene Gruppen-Management-Konzept ermöglicht, innerhalb des Systems kollaborative Gruppen zu erstellen. Ein Gruppenmitglied kann eine kollaborative Gruppe betreten, wenn es dazu berechtigt ist. Ein Gruppenmitglied hat die Möglichkeit, seine Gruppe zu verlassen, wenn es nicht mehr zu dieser Gruppe gehören will.

- **Bearbeitung von Dokumenten**
  Das System erfüllt alle an die Dokument-Verwaltung gestellten Anforderungen: Neue Dokumente können im System innerhalb einer kollaborativen Gruppe erzeugt werden. Die einzelnen Abschnitte eines Dokumentes können unterschiedlichen Gruppenmitgliedern zur Bearbeitung zugewiesen werden. Notizen und Anmerkungen können an die Abschnitte des Dokumentes angefügt werden. Die Dokument-Versionen können synchronisiert werden.

- **Rechte-Management**
  Ein detailliertes Konzept der Rechte-Verwaltung wurde in dieser Arbeit nicht ausgearbeitet, da es ein sehr umfangreiches Thema ist und seine Entwicklung den Rahmen dieser Arbeit überschreiten würde. Dennoch wurden ein Modul für das Rechte-Management vorgesehen und eine Schnittstelle definiert, die die Möglichkeit für eine spätere Nach-Implementierung offen lässt. Darüber hinaus wurde eine allgemeine Strategie zur Editierrechte-Vergabe innerhalb eines Dokumentes ausgearbeitet.

Der Entwurf genügt den im Kapitel „Grundlagen“ definierten Eigenschaften eines kollaborativen Editiersystems (Abschnitt 2.3.2) wie folgt:

- Kooperationsunterstützung
  Das System unterstützt die Zusammenarbeit an Dokumenten. Jedem Team-Mitglied steht das gemeinsame Arbeitsobjekt (das Dokument) zur Verfügung. Das System unterstützt die Verteilung der Aufgaben innerhalb einer Gruppe und vermittelt dem Benutzer, welche Teile des

Dokumentes zu seinem Aufgabenbereich gehören und welche den anderen Gruppenmitgliedern.

- Kollaborative Awareness
  Das System ermöglicht den Benutzern die Wahrnehmung der anderen Teilnehmer durch die Überwachung des Netzes (dem Benutzer wird angezeigt, welche Gruppenmitglieder zur Zeit online sind). Der Benutzer kann die Arbeitsergebnisse der anderen Teilnehmer ansehen, um besseres Verständnis für die gemeinsame Aufgabe zu haben. Die Benutzer können an Teile des gemeinsamen Dokumentes Notizen (Annotationen) anbringen, die ermöglichen, die Qualität des Dokumentes zu steigern.

- Nebenläufigkeitskontrolle
  Im System wird keine Nebenläufigkeitskontrolle eingesetzt, da den nebenläufigen Zugriffen auf die gemeinsamen Arbeitsobjekte durch feingranulare Editierrechtevergabe vorgebeugt wird.

- Versionierung, Annotationen
  Das System unterstützt keine Versionierung der Dokumente. Die Historie der Bearbeitung kann nur durch die expliziten Vorkehrungen des Benutzers geführt werden, z.B. durch das Hinzufügen von bestimmten Annotationen oder durch das Sicherungs-Speichern der Inhalte außerhalb des Editiersystems.

- Zugriffskontrolle
  Das System unterstützt die Vergabe von Zugriffsrechten.

- Benutzerverwaltung
  Das System unterstützt die Verwaltung von Gruppen und Gruppenmitgliedern.

- Konflikterkennung
  Das System ist im Stande, zwei Versionen eines Dokumentes zu vergleichen, die Unterschiede zu erkennen und die Versionen zusammenzuführen. Durch die entworfene Editierrechte-Strategie ist eine Konfliktsituation bei der Zusammenführung von zwei Versionen eines Dokumentes praktisch unwahrscheinlich. Nur wenn das Editierrecht eines Gruppenmitglieds aufgehoben und einem anderen Gruppenmitglied vergeben wurde, kommt es zu einem Konflikt. In diesem Fall wird der Benutzer darauf hingewiesen und kann die notwendigen Schritte einleiten, um seine Änderungen zu sichern.

Wie diese durchgeführte Analyse gezeigt hat, unterstützt das entwickelte Konzept fast alle wichtigen Eigenschaften eines kollaborativen Editiersystems. Der realisierte Prototyp hat seinerseits gezeigt, dass dieses Konzept für den Einsatz in dynamischen Umgebungen für spontane kollaborative Arbeit an Dokumenten tragfähig ist.

Obwohl das entworfene Konzept den gestellten Anforderungen genügt, existieren einige Aspekte, die im Design noch nicht bedacht wurden, aber für die Vervollständigung des kollaborativen Editiersystems wünschenswert erscheinen.

Dazu gehört in erster Linie ein Sicherheitskonzept, das die Integrität und die Vertraulichkeit der Dokumente, Authentifizierung der Benutzer und sichere Kommunikation garantieren wird. Das kann unter anderem durch den Einsatz von Passwörtern und TAN-Systemen, Zertifikaten und Verschlüsselungs-Verfahren [Mund] erreicht werden.

Des Weiteren braucht das System ein Versionierungs- oder ein Historie-Konzept für Dokumente. Das wird benötigt, um den Verlauf der Dokumentbearbeitung für die Gruppenmitglieder besser nachvollziehbar zu gestalten. Außerdem wird dadurch dem Benutzer die Möglichkeit gegeben, falsche oder nicht mehr erwünschte Änderungen rückgängig zu machen. Für diese Aufgabe könnte z.B. eines der schon existierenden Versionierungstools genommen und ins Editiersystem integriert werden. Da das Editiersystem keine zentrale Instanz hat, könnte ein solches Tool nur lokal bei jedem Benutzer laufen. Das Editiersystem würde dafür sorgen, dass die Änderungen an Dokumenten beim Speichern auch unter Versionskontrolle gestellt werden. Für die Wahl eines Tools ist in diesem Fall die Existenz einer annehmbaren Schnittstelle ausschlaggebend.

Als Letztes kann hier ein Werkzeug für die zusätzliche Unterstützung der Kommunikation zwischen den Gruppenmitgliedern erwähnt werden. Da das Editiersystem auch eine starke Verteilung der Gruppe unterstützen soll, können nur systeminterne Kommunikationsmechanismen unter Umständen nicht ausreichend sein, z.B. für die Absprache der nächsten Schritte in Bezug auf die Bearbeitung des Dokumentes oder beim Verschieben des End-Termins der Fertigstellung des Dokumentes usw. Die Unterstützung einer expliziten Kommunikation ist in solchen Fällen hilfreich. Dazu könnte z.B. ein Chat-Werkzeug ins System integriert werden oder das System um einen E-Mail-Service erweitert werden.

Kapitel 5

Zusammenfassung und Ausblick

5.1 Zusammenfassung

Ziel dieser Masterarbeit war die Entwicklung eines kollaborativen Editiersystems, das die Bearbeitung von strukturierten Dokumenten in einer Gruppe asynchron und dezentral unterstützt. Dieses Ziel wurde erreicht. Für gemeinsame Dokumente wurde ein spezielles Format entworfen, das sowohl kollaborative Informationen als auch fachliche Inhalte der Dokumente umfasst. Außerdem ermöglicht es eine fein-granulare Vergabe von Editierrechten. Das System unterstützt eine replizierte Haltung der gemeinsamen Dokumente und ihre Synchronisierung. Basierend auf einer Peer-to-Peer-Architektur und einer ausgearbeiteten Rechtevergabe-Strategie, die nebenläufigen Zugriffen vorbeugt, ermöglicht das System eine spontane Dokumenten-Erstellung in einer kollaborativen Gruppe unter von Bedingungen der dynamischen und mobilen Umgebungen.

Diese Arbeit hat gezeigt, dass ein Peer-to-Peer-Konzept für die kollaborative Arbeit in dynamischen Umgebungen tragfähig ist. Mit den passenden Strategien für die Rechtevergabe und Datenhaltung ist eine asynchrone Erstellung von Dokumenten realistisch und produktiv. Allerdings muss beachtet werden, dass dieses Konzept sich eher für kleine Gruppen und relativ kurzfristige Aufgaben eignet. Der Peer-to-Peer-Ansatz kann bei einer großen Anzahl an Beteiligten zu einem Kommunikations-Overhead führen. Darüber hinaus sind gemeinsame Arbeitsfortschritte der Gruppe von der Präsenz der einzelnen Gruppenmitglieder abhängig, da es keine zentrale Stelle gibt, wo die einzelnen Ergebnisse gesammelt und zusammengeführt werden. Dementsprechend sind das Interesse der Gruppenmitglieder an einer produktiven Zusammenarbeit und ihre Initiative stark gefragt. Für die Bearbeitung einer

langfristigen Aufgabe ist ein zentralisiertes System zu empfehlen, da sich der durch das Aufsetzen der Infrastruktur entstehende organisatorische, zeitliche und Ressourcen-Aufwand bei langfristigen Projekten auszahlt. Wenn aber eine leichtgewichtige Lösung für spontane Kollaboration gefragt ist, ist der in dieser Arbeit vorgestellten Ansatz empfehlenswert.

In Rahmen dieser Arbeit wurde ein Beitrag zur CSCW-Vision gemacht, den Menschen die gemeinsame Arbeit durch eine umfassende und an die Bedürfnisse und Umgebung ausgerichtete Computerunterstützung zu ermöglichen und zu erleichtern.

5.2 Ausblick

Die Entwicklungen der letzten Zeit zeigen, dass die Verbreitung und die Popularität der Kooperations-unterstützenden Werkzeuge auf Text-Basis steigen. Wenn noch vor einigen Jahren kollaborative Textbearbeitungssysteme im Rahmen von Forschungsprojekten entwickelt wurden, werden jetzt zunehmend kommerzielle Lösungen angeboten. Das beste Beispiel dafür ist Google Docs [Google Docs]. Dieses System unterstützt die Online-Bearbeitung von Dokumenten, Tabellen und Präsentationen, ihr lokales Abspeichern in gängigen Formaten und das Hochladen der lokal existierenden Dokumente. Außerdem ermöglicht das System die Zusammenarbeit mit anderen Personen an den Dokumenten. Diese Entwicklung zeigt, dass ein Bedarf an solchen Werkzeugen existiert. Die Tendenz geht dabei in die Richtung eines kombinierten Online-/Offline-Betriebes. Die meisten aktuellen Systeme, wie z.B. Google Docs, basieren auf einer zentralisierten Lösung. Welche Art der Computer-Unterstützung der kooperativen Arbeit sich durchsetzt und gebraucht wird, wird sich erst in der Zukunft herausstellen. Wichtig ist, bei der Entwicklung von solchen Werkzeugen offene Standards einzusetzen und das Design flexibel in Bezug auf die Erweiterbarkeit des Systems und die Austauschbarkeit der Komponenten zu gestalten, um so die Werkzeuge anpassungsfähig und lebensfähig hinsichtlich der kommenden Trends zu gestalten.

Literaturverzeichnis

[Bisignano] Bisignano, Mario, G. D. Modica und O. Tomarchio. *A JXTA Compliant Framework for Mobile Handheld Devices in Ad Hoc Networks*. In: *ISCC '05: Proceedings of the 10th IEEE Symposium on Computers and Communications*, S. 582–587, Los Alamitos, CA, USA, 2005. IEEE Computer Society. ISBN: 0-7695-2373-0. URL: http://doi.ieeecomputersociety.org/10.1109/ISCC.2005.12.

[Borghoff] Borghoff, Uwe M. und J. H. Schlichter. *Rechnergestützte Gruppenarbeit - Eine Einführung in verteilte Anwendungen*. Springer, Berlin, 1998. ISBN: 3-540-62873-8.

[BSCW] BSCW. *Basic Support for Cooperative Work*. GMD Frauenhofer FIT, 2006. URL: http://bscw.fit.fraunhofer.de. Stand: Juli 2007.

[Caituiro-Monge] Caituiro-Monge, Hillary, K. Almeroth und M. del Mar Alvarez-Rohena. *Friend relay: a resource sharing framework for mobile wireless devices*. In: *WMASH '06: Proceedings of the 4th international workshop on Wireless mobile applications and services on WLAN hotspots*, S. 20–29, Los Angeles, CA, USA, 2006. ACM Press. ISBN: 1-59593-470-7. URL: http://doi.acm.org/10.1145/1161023.1161027.

[DocBook] DocBook. *Offizielle Homepage von DocBook*. O'Reilly & Associates, Inc. URL: http://www.docbook.org. Stand: August 2007.

[Dustdar] Dustdar, Schahram, H. Gall und M. Hauswirth. *Saftware-Architekturen fü Verteilte Systeme*. Springer, Berlin, Heidelberg, 2003. ISBN: 3-540-43088-1.

[Ebersbach] Ebersbach, Anja, M. Glaser und R. Heigl. *WikiTools*. Springer, Berlin, Heidelberg, 2005. ISBN: 3-540-22939-6.

[Eckert] Eckert, Claudia. *IT-Sicherheit: Konzepte - Verfahren - Protokolle*. Oldenbourg Wissenschaftsverlag GmbH, München, 2004. ISBN: 3-486-20000-3.

[Eclipse] Eclipse. *An open development platform*. The Eclipse Foundation, 2008. URL: http://www.eclipse.org. Stand: März 2008.

[Ellisa] Ellis, C. A. und S. J. Gibbs. *Concurrency control in groupware systems*. In: *SIGMOD '89: Proceedings of the 1989 ACM SIGMOD international conference on Management of data*, S. 399–407, Portland, Oregon, United States, 1989a. ACM Press. ISBN: 0-89791-317-5. URL: http://doi.acm.org/10.1145/67544.66963.

[Ellisb] Ellis, C.A., S. Gibbs und G. Rein. *Groupware: Some Issues and Experiences*. Communications of the ACM, 34, ACM Press, 1991b.

[Fontaine] Fontaine, Robin La und N. Whitaker. *A Generalized Grammar for Three-way XML Synchronization*. In: *XML '05: Proceedings of the XML Conference by RenderX*, Atlanta, Georgia, U.S.A., 2005. RenderX. URL: http://www.idealliance.org/proceedings/xml05/abstracts/paper102.HTML.

[Gamma] Gamma, Erich, R. Helm, R. Johnson und J. Vlissides. *Entwurfmuster*. Addison-Wesley, München, 2004. ISBN: 3-8273-2199-9.

[Gerlicher] Gerlicher, Ansgar. *Erweiterung bestehender Anwendungen um kollaborative Funktionen mit Hilfe des Collaborative Editing Framework for XML (CEFX)*. Aktuelle Trends in der Softwareforschung, 2:150–165, 2004.

[Google Docs] Google Docs. *The Web Word Processor*. Google, 2006. URL: http://docs.google.com. Stand: Juli 2007.

[Gross] Gross, Tom und M. Koch. *Computer-Supported Cooperative Work*. Oldenbourg Wissenschaftsverlag GmbH, München, 2007. ISBN: 978-3-486-58000-6.

[Härder] Härder, Theo. *Verteilte und Parallele Datenbanksysteme*. Universität Kaiserslautern Fachbereich Informatik, SS 2002, 2007. URL: http://wwwdvs.informatik.uni-kl.de/courses/VDBS/SS2002/Vorlesungsunterlagen/Kapitel.07.pdf. Stand: Juni 2007.

[Hodel] Hodel, Thomas B. und K. R. Dittrich. *Concept and prototype of a collaborative business process environment for document processing*. Data & Knowledge Engineering, 52:61–120, Elsevier B.V., 2004.

[Holzinger] Holzinger, Andreas. *Basiswissen IT/Informatik, Band 3: Internet und WWW*. Vogel, Würzburg, 2004. ISBN: 3-8023-1899-4.

[ICQ] ICQ. *ICQ Instant Messaging Program*. AOL. URL: http://www.icq.com. Stand: Juli 2007.

[Ignat] Ignat, C. und M. Norrie. *Tree-based Model Algorithm for Maintaining Consistency in Real-Time Collaborative Editing Systems*. In: *The Fourth International Workshop on Collaborative Editing Systems*, 2002. ACM Press.

[IT Wissen] IT Wissen. *Das grosse Online-Lexikon für Informationstechnologie*. DATACOM Buchverlag GmbH, 2004–2007. URL: http://www.itwissen.info. Stand: September 2007.

[JDOM] JDOM. *Java-based solution for accessing, manipulating, and outputting XML data from Java code*. jdom.org. URL: http://www.jdom.org. Stand: März 2008.

[JXME] JXME. *JXTA Java Micro Edition Project, offizielle Webseite*. URL: http://jxme.jxta.org. Stand: Juni 2007.

[JXSE] JXSE. *Java-Referenz-Implementierung von JXTA*. Sun Microsystems, Inc. URL: https://jxta-jxse.dev.java.net. Stand: März 2008.

[JXTA] JXTA. *Spezifikation & Referenz-Implementierung, offizielle Webseite*. URL: http://www.jxta.org. Stand: Juni 2007.

[JxtaProgGuide] JxtaProgGuide. *JXTA v2.3.x, Java Programmer's Guide*. Sun Microsystems, Inc., 2005. URL: http://www.jxta.org/docs/JxtaProgGuide_v2.3.pdf. Stand: Juni 2007.

[Kampffmeyer] Kampffmeyer, Ulrich. *Dokumententechnologien: Wohin geht die Reise?*. Project Consult GmbH, 2003. ISBN: 3-9806756-4-5.

[Kemper] Kemper, Alfons und A. Eickler. *Datenbanksysteme: Eine Einführung*. Oldenbourg Wissenschaftsverlag GmbH, München, 2004. ISBN: 3-486-27392-2.

[Lamport] Lamport, Leslie. *Das LaTeX-Handbuch*. Addison-Wesley, München, 1995. ISBN: 3-89319-826-1.

[Leone] Leone, Stefania, T. B. Hodel und H. Gall. *Concept and architecture of an pervasive document editing and managing system*. In: *SIGDOC '05: Proceedings of the 23rd annual international conference on Design of communication*, S. 41–47, Coventry, United Kingdom, 2005. ACM Press. ISBN: 1-59593-175-9. URL: http://doi.acm.org/10.1145/1085313.1085326.

[Lindholm] Lindholm, Tancred, J. Kangasharju und S. Tarkoma. *Fast and simple XML tree differencing by sequence alignment*. In: *DocEng '06: Proceedings of the 2006 ACM symposium on Document engineering*, S. 75–84, Amsterdam, The Netherlands, 2006. ACM Press. ISBN: 1-59593-515-0. URL: http://doi.acm.org/10.1145/1166160.1166183.

[Maibaum] Maibaum, Nico und T. Mundt. *JXTA: A Technology Facilitating Mobile Peer-To-Peer Networks*. In: *MOBIWAC '02: Proceedings of the International Workshop on Mobility and Wireless Access*, S. 7–13, Forth Worth, Texas, USA, 2002. IEEE Computer Society. ISBN: 0-7695-1843-5.

[Mascolo] Mascolo, Cecilia, L. Capra und W. Emmerich. *An XML-based Middleware for Peer-to-Peer computing*. In: *P2P '01: Proceedings of the First International Conference on Peer-to-Peer Computing*, S. 69–74, Linköping, Sweden, 2001. IEEE Computer Society. ISBN: 0-7695-1503-7. URL: http://doi.ieeecomputersociety.org/10.1109/P2P.2001.990428.

[Mattern] Mattern, Friedemann. *Pervasive Computing/ Ubiquitous Computing*. Informatik-Spektrum, 24(3):145–147, 2001.

[MediaWiki] MediaWiki. *Offizielle Homepage von MediaWiki, der Wiki-Software*. URL: http://www.mediawiki.org. Stand: September 2007.

[Michael] Michael, Dirk. *Entwurf von P2P-Anwendungen mit Hilfe der JXTA-Technologie*. Ausarbeitung, Technische Universität Ilmenau, Fakultät für Informatik und Automatisierung, Fachgebiet Rechnerarchitekturen, 2003.

[Mintert] Mintert, Stefan. *Leise Revolution*. iX Magazin für Professionelle Informationstechnik, 7:126–137, Heise-Verlag, 1995.

[MSP2P] MSP2P. *MS PeerToPeer: Windows Peer-to-Peer Networking*. Microsoft TechNet, 2006. URL: http://www.microsoft.com/technet/network/p2p/default.mspx. Stand: Juni 2007.

[Mund] Mund, Horst. *Berechtigungsstrukturen in kollaborativen Umgebungen*. Diplomarbeit, Hochschule für Angewandte Wissenschaften Hamburg, Fakultät Technik und Informatik, Studiendepartment Informatik, 2006.

[Neyem] Neyem, Andrés, S. F. Ochoa und J. A. Pino. *A Strategy to Share Documents in MANETs using Mobile Devices*. In: *ICACT '06: Proceedings of the 8th International Conference on Advanced Communication*

Technology, S. 1400–1404, Phoenix Park, Korea, 2006. IEEE Press. ISBN: 89-5519-129-4.

[OASIS DocBook TC] OASIS DocBook TC. *The OASIS DocBook Technical Committee*. OASIS, 1993–2007. URL: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=docbook. Stand: September 2007.

[Roussev] Roussev, Vassil, G. P. Priego und G. G. I. Richard. *TouchSync: Lightweight Synchronization for Ad-Hoc Mobile Collaboration*. In: *CTS '06: Proceedings of the International Symposium on Collaborative Technologies and Systems*, S. 181–188, Las Vegas, NV, USA, 2006. IEEE Computer Society. ISBN: 0-9785699-0-3. URL: http://dx.doi.org/10.1109/CTS.2006.67.

[Schlichter] Schlichter, Johann H. *Computergestützte Gruppenarbeit*. Institut für Informatik, Technische Universität München, WS 2001/02, 2001. URL: http://www11.informatik.tu-muenchen.de/lehre/lectures/ws2001-02/cscw/extension/latex/cscw_course-student.pdf. Stand: Juli 2007.

[Schwabe] Schwabe, Gerhard, N. Streitz und R. Unland. *CSCW-Kompendium*. Springer, Berlin, 2001. ISBN: 3-540-67552-3.

[Shi] Shi, Hao, Y. Zhang, J. Zhang, E. Beal und N. Moustakas. *Collaborative Peer-to-Peer Service for Information Sharing Using JXTA*. In: *IMSCCS '06: Proceedings of the First International Multi-Symposiums on Computer and Computational Sciences - Volume 1*, S. 552–559, Hangzhou, China, 2006. IEEE Computer Society. ISBN: 0-7695-2581-4. URL: http://doi.ieeecomputersociety.org/10.1109/IMSCCS.2006.45.

[SWT] SWT. *The Standard Widget Toolkit*. eclipse.org. URL: http://www.eclipse.org/swt. Stand: März 2008.

[Teufel] Teufel, Stephanie, C. Sauter, T. Mühlherr und K. Bauknecht. *Computerunterstützung für die Gruppenarbeit*. Addison-Wesley, 1995. ISBN: 3-89319-878-4.

[Trieloff] Trieloff, Lars. *DocBook-XML*. mitp-Verlag, Bonn, 2005. ISBN: 3-8266-1519-0.

[TWiki] TWiki. *The Open Source Wiki for the Enterprise*. URL: http://twiki.org/. Stand: September 2007.

[Wiese] Wiese, Jens. *Einsatzgebiete von nativen XML-Datenbanken*. Diplomarbeit, Hochschule für Angewandte Wissenschaften Hamburg, Fachbereich Medientechnik, 2004.

[Windows Life] Windows Life. *Windows Life Messanger*. Microsoft, 2006. URL: http://www.windowslivediscovery.de/wld.asp. Stand: Juli 2007.

[WinEdt] WinEdt. *Editor für [La]Tex Dokumente*. Aleksander Simonic, 1993–2008. URL: http://www.winedt.com. Stand: März 2008.

[Wong] Wong, Raymond K. *Collaborative Hypertext Editing in Mobile Environment*. In: *MMM '04: Proceedings of the 10th International Multimedia Modelling Conference*, S. 300–307, Brisbane, Australia, 2004. IEEE Computer Society. ISBN: 0-7695-2084-7.

[XML] XML. *Extensible Markup Language*. URL: http://www.w3.org/XML. Stand: Juni 2007.

[XPath] XPath. *XML Path Language, W3C Recommendation*. W3C, 1999. URL: http://www.w3.org/TR/xpath. Stand: März 2008.

[Yang] Yang, Yun, C. Sun, Y. Zhang und X. Jia. *Real-Time Cooperative Editing on the Internet*. IEEE Internet Computing, 04(3):18–25, IEEE Computer Society, 2000. URL: http://doi.ieeecomputersociety.org/10.1109/4236.845386.

[Yushun] Yushun, Li und S. MeiLin. *Mobile Collaboration Writing*. In: *PIMRC '03: Proceedings of the 14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, S. 849–853, Beijing, China, 2003. IEEE Press. ISBN: 7-5053-5066-8.

[ZeroConf] ZeroConf. *Zero Configuration Networking*. The IETF Zeroconf Working Group. URL: http://www.zeroconf.org. Stand: Juli 2007.

[Zivadinovic] Zivadinovic, Dusan. *Vermittlungsprotokoll: Komfort im LAN mit Bonjour*. Heise Netze, 2006. URL: http://www.heise.de/netze/artikel/77938. Stand: Juni 2007.

Anhang A

Inhalt der CD

Die beiliegende CD enthält den Source-Code der entwickelten Anwendung *Kollaborative Editor* mit allen für die Implementierung benutzen Bibliotheken, unter anderem JXTA-Implementierung, SWT-Pakete, JDOM-API usw. Der Source-Code befindet sich im Verzeichnis mit dem Namen „CollaborativeEditor“. Darüber hinaus beinhaltet das gleiche Verzeichnis eine Standard-Konfiguration der JXTA-Komponente.

Des Weiteren befindet sich auf der CD das XML-Schema für das kollaborative Dokument, die Versions-Schema und das Identifikatoren-Schema.

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, den	
Ort, Datum	Unterschrift